Project

Simulation

Group Name: Team TAE (Trial and Error)

Team Leader: Cris Gievid M. Narca

CI/CD: Anjhillian S. Cabrera

Mari Cris P. Ayes

Developer: Joey Ann O. Robles

Mari Cris P. Ayes

Documentation: Pauleen S. Pineda

**Table of Contents**

CHAPTER 1 – INTRODUCTION

1.1 BACKGROUND OF THE STUDY

1.2 OBJECTIVE

1.3.1 General Objectives

1.3.2 Specific Objectives

CHAPTER 2 – RESEARCH METHODOLOGY

2.1 CONCEPTUAL FRAMEWORK

2.2 SOFTWARE DIAGRAM

2.3 SOFTWARE CONSIDERATIONS

2.3.1 GitLab

2.3.2 Jenkins

2.3.3 Maven

2.3.4 Sonarqube

2.3.5 Ansible

2.3.6 Apache Tomcat

2.3.7 Selenium

2.3.8 Amazon Web Services

2.3.9 Putty

2.3.10 Docker

2.4 WORKING THEORIES

2.4.1 Creating Stacks in AWS

2.4.2 DevOps Platform

2.4.3. Jenkins

2.4.4 Maven Configuration

2.4.5 Sonarqube Configuration

2.4.6 Ansible Configuration

2.4.7 Selenium Configuration

2.4.8 GitLab

2.4.9 Pipeline

2.4.10 Java based Web Application in Tomcat

**CHAPTER 1 – INTRODUCTION**

**1.1 BACKGROUND OF THE STUDY**

As innovation accelerates and customer’s needs rapidly evolve, businesses must become increasingly agile. But unlike the previous system, development always have issues in collaborating with production which causes slow growth in production, more risks to errors, cost, rework which is very time consuming and mostly likely inefficient. Thus, business goals and demands to the customers were not achieved.

In order to meet the demands of an agile business, IT operations need to deploy applications in a consistent, repeatable, and reliable manner. This can only be fully achieved with the adoption of automation. DevOps is a new term that focuses on improved collaboration, communication, and integration between software developers and IT operations. It is simply the collaboration of development and operation that produce a continuous integration and continuous deployment of codes to production. It basically automates the whole lifecycle of the system, from build, test and to the deploy process of the system. Some of tools used in Devops are GitLab, Jenkins, Maven, Sonarqube, Ansible, Tomcat and Selenium.

The primary goal of DevOps is to automate the whole process that will prevent risk in errors, rework, and higher operating expense. It will improved deploy frequency which can lead to faster time to market. It aims to deliver higher quality of software and lower the failure rate to end users at a faster pace and will shortened the lead time that will eventually increase production, faster mean time to recovery  
and revenue of the business.

**1.2 OBJECTIVE**

**1.3.1 General Objectives**

The main objective of the study is to create a system that will automate development, testing and deployment of codes to production.

**1.3.2 Specific Objectives**

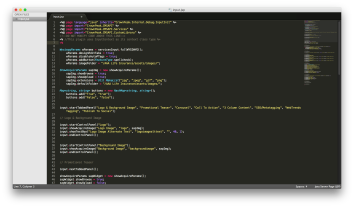
* To create a java based application that will be used to test the system
* To integrate different tools such as GitLab, Maven, Sonarqube, Ansible, Selenium and Tomcat to Jenkins
* To produce continuous integration and continuous deployment
* To maximize the efficiency of the whole system

**CHAPTER 2 – RESEARCH METHODOLOGY**

**2.1 CONCEPTUAL FRAMEWORK**

**Developers**



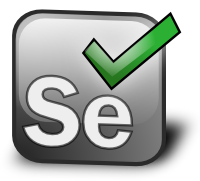
****

****

**CI/CD**

****

****

****



**CLIENT**

**Figure 2.1.** *Conceptual Diagram of the System*

From Figure 1, the developers created a web-based java project and deployed their codes to the repository. The working repository of the system is GitLab. Jenkins will then pull the code from the repository. Jenkins will enable the system to automatically process the code from different tools. Maven will build the code and deploy it to sonarqube to assess whether the code submitted passed the quality gates set by the client. The processed project will be deployed to ansible. This tool consists of a playbook that will build the docker image of the project and will then be deployed in Tomcat web-server. The output will be tested in selenium to assess whether the application is passed the user acceptance test.

**2.2 SOFTWARE DIAGRAM**

N

Y

Y

N

Start

Set-up a Stack in a given AWS Instance

Load the given JSON file to have a DevOps Platform

Create a GitLab account

Find/Make a Java WebApp Project

Push the Project into the Repository in Gitlab

Configure Maven Build Job in Jenkins

Configure Sonarqube Test Job in Jenkins

Is Maven build successful?

Is Sonarqube

Test successful?

N

Configure Selenium Test Job in Jenkins

Configure Ansible Deployment Job in Jenkins

Is Deployment successful?

Y

N

End

Is Selenium Test successful?

Y

**Figure 2.2.** *Software Design Flowchart*

Figure 2.2 shows the software diagram of the system. First, stack is created in a given AWS (Amazon Web Services) instance. Stack is then configured (see Figure 2.3 to 2.8) which includes the template, tags, instance type, instance size, etc. Then, the JSON file is loaded in order to come up with a DevOps Platform (see Figure 2.9) which is addressed to <http://54.68.198.28/>. After DevOps platform is established, GitLab account is created which will serve as working repository of the system. A java based web application is then pushed to the repository. In order to connect GitLab to Jenkins, web hook is used. Once GitLab is connected to Jenkins, the maven job is created in Jenkins and is configured from general down to the post build action (see Figure 2.11 to 2.16). Then, maven job is build. If maven build is not successful, return to maven job configuration and resolve the errors, but if it is successful, it will continue with test job. Test job serves as sonarqube. It is created and configured in Jenkins (see Figure 2.17 to 2.21). Sonarqube will test the quality of the codes pushed and will display the summary of the quality of codes (see Figure 2.22). If the code passed the quality gates set by the clients, it will proceed to ansible deployment job. Ansible deployment job is created and configured in Jenkins (see Figure 2.23 to 2.27). It will build the docker image that will be deployed to tomcat web server. Once the web application appears on the Tomcat web server, it will proceed to selenium. Selenium test job is created and configured in Jenkins. Selenium will test whether the web application is properly working.

In order to display the status of the whole system, a pipeline is created. Stages

It shows whether the job build is successful or not. It is useful for error detection, ……. **SOMETHING**

**2.3 SOFTWARE CONSIDERATIONS**

**2.3.1 GitLab**

GitLab is an open-source code collaboration platform that is responsible for hosting and visualizing Git repositories. It enables developers to create, review, track issues, and deploy code bases. GitLab was written in Ruby and it was officially incorporated in 2014 and operated in San Francisco, California.

It gives a complete control over the repositories and allows the user to decide whether they are public or private for free .One of the major benefits of the system is the version control that enables you to go back or recover certain file from the previous commit. It is also a convenient user interface that enables users to access everything from one screen to another, anytime and anywhere.

**2.3.2 Jenkins**

Jenkins is a free application that allows continuous integration and continuous delivery of projects, regardless of the platform used. It can handle any kind of build, push files to various artifact repositories, integrate tools for testing and deployment. Jenkins is a java based continuous build system.It is the primary platform for plugins, whereas it is supported by over 400 plugins. .These plugins cover everything from version control systems, build tools, code quality metrics, build notifiers, integration with external systems, UI customization and much more.

Jenkins is used for projects in a wide variety of languages and technologies, including .NET, Ruby, Groovy, Grails, PHP and more, as well as Java. It is also extremely flexible and easy to adapt to its assigned purpose and its user interface is simple, easy to use, intuitive, and visually appealing.

**2.3.3 Maven**

Maven is a project management and comprehension tool. It provides developers a complete build lifecycle framework whereas the development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle. Maven makes life of developer easy while creating reports, checks, build and testing automation setups.

Maven's primary goal is to allow a developer to comprehend the complete state of a development effort in the shortest period of time. In order to attain this, Maven attempt to the build process easy, provide a uniform build system; provide quality project information; provide guidelines for best practices development and allows transparent migration to new features. One advantage of Maven is that it has better dependency management. It can add new dependencies quickly, identify unused and transitive dependencies, and it can create reports showing all dependencies used on a project. Second, Maven’s default plugins and life cycle allow a project to perform common build actions with touching build configuration file. Third, it has better debugging and collaboration. Maven repositories allow an artifact’s source code and Javadoc to be published along with the artifact’s JAR. Fourth, it has a consistent project structure which makes it easier to understand each project. And lastly, Maven projects can be use POM (project object model) hierarchy to reduce the duplication.

**2.3.4 Sonarqube**

Sonarqube is an open platform for code quality management where developers can manage, track and improve the quality of the source code. It is written in java but it can analyze code in 20 different programming languages like; Java (including Android), C/C++, Objective-C, C#, PHP, Flex, Groovy, JavaScript,Python, PL/SQL, COBOL, Swift and many more. It covers the 7 axes of code quality such as; architecture and design, duplications, unit tests, complexity, potential bugs, coding rules and comments. It is expandable with the use of plugins.

Sonarqube platform enables automatic detection of bugs and provides an opportunity to fix them before rolling software out to production. It provides fully automated analysis; it integrates with continuous integration tools like Jenkins. Thus, it avoids risks of software development within a short period of time, improves quality and productivity.

**2.3.5 Ansible**

Ansible is a configuration management, deployment and orchestration tool that provides an automated infrastructure for managing systems devices and applications. It is written in Python and provides an agentless approach that focuses on management of the destination device and application over SSH. Ansible implements playbooks and modules. Playbooks are series of tasks to be executed on a host and provide the fundamental workflow. On the other hand, Modules perform the operations based on the directives of the tasks and playbooks.

Ansible’s simplicity and maximum ease of use that can configure systems, deploy software, and orchestrate tasks such as continuous deployments. It aims to be clear (uses a simple syntax – YAML, easier to understand), fast (fast to learn, fast to set up), complete (it does three things in one), efficient (no extra software on your servers) and lastly secure (it uses ssh).

**2.3.6 Apache Tomcat**

Apache Tomcat is open source software that implements java Servlet and Java Server pages, enabling sites to run Java servlets and Java-based dynamic content. It allows developers to effortlessly build and check today’s connected Web applications. The advantages of apache tomcat are flexibility and stability.

**2.3.7 Selenium**

Selenium is a portable software testing framework for web applications. It can be deployed on Windows and Linux. It is a set of tools that supports rapid development of test automation for web-based applications. It provides a rich set of testing functions specifically geared to the needs of testing of a web application. These operations are highly flexible, allowing many options for locating UI elements and comparing expected test results against actual application behavior. The tests are written as HTML tables or coded in a number of popular programming languages. It can be run directly in most modern web browser.

Selenium is composed with three components, each one of these components are responsible for the reliable automated testing. These components are; Selenium IDE – it has a recording feature, which will keep account of user actions as they are performed and store them as a reusable script to play back. Selenium Remote Control (RC) – it allows the developer to use a programming language to maximized the flexibility and extensibility in developing the test logic and lastly Selenium Grid – it allows the rc to run the test suites in different environment. It can run in different operating system and browser configuration. Some of the advantages of Selenium are frequent regression testing, rapid feedback to developers, virtually unlimited iterations of test case execution and can easily find defects missed by manual testing.

**2.3.8 Amazon Web Services**

Amazon Web Services (AWS) offers a suite of cloud-computing services that make up an on-demand computing platform. These services operate from 13 geographical regions across the world. It offers more than 70 services, spanning a wide range, including compute, storage, networking, database, analytics, application services, deployment, management, mobile, developer tools and tools for the Internet of things, these includes EC2 (Amazon Elastic Compute Cloud) and S3 (Amazon Simple Storage Service).

AWS as the leading cloud computing provider ensure that their cloud model is composed of these characteristics; this model should be on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service – this allows the consumer unilaterally provision computing capabilities such as network storage, all its capabilities are available through the network and can be accessed easily. It is more convenient, faster and more reliable.

**2.3.9 Putty**

Putty is a free and open-source terminal emulator; serial console and network file transfer application. It does not have the same functionality as WinSCP or Tunnelier It was originally written for Microsoft Windows, but it has been ported to various other operating systems. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port. Official ports are available for some Unix-like platforms, with work-in-progress ports to Classic Mac OS and Mac OS X, and unofficial ports have been contributed to platforms such as Symbian Windows Mobile and Windows Phone.

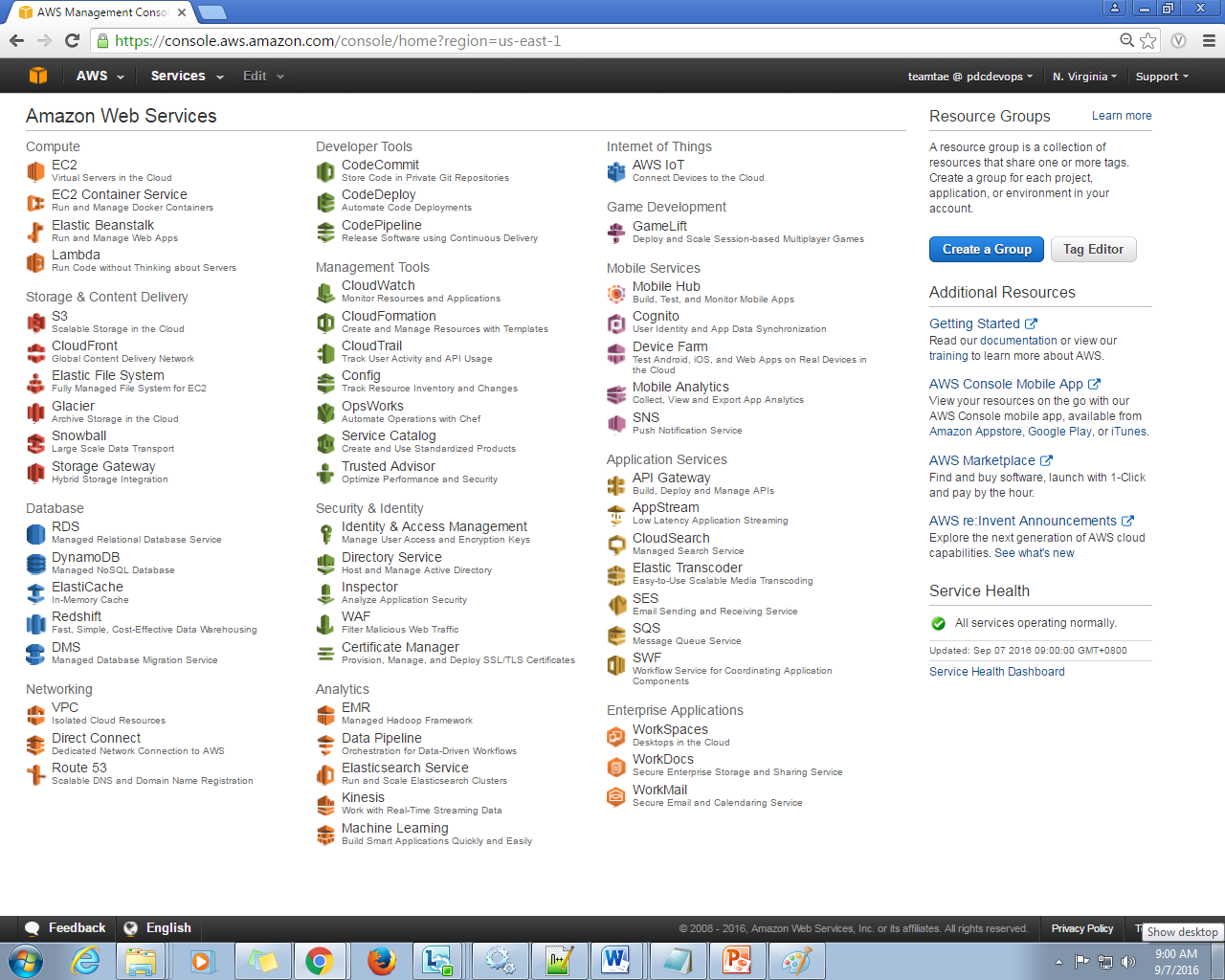
**2.3.10 Docker**

Docker is an open-source engine that automates the deployment of applications into containers. It adds an application deployment engine on top of a virtualized container execution environment. It is designed to provide a lightweight and fast environment in which to run the code as well as an efficient workflow to get that code from one source to the designated test environment and then into production. Docker made an easy and lightweight way to model reality, fast and efficient development life cycle.

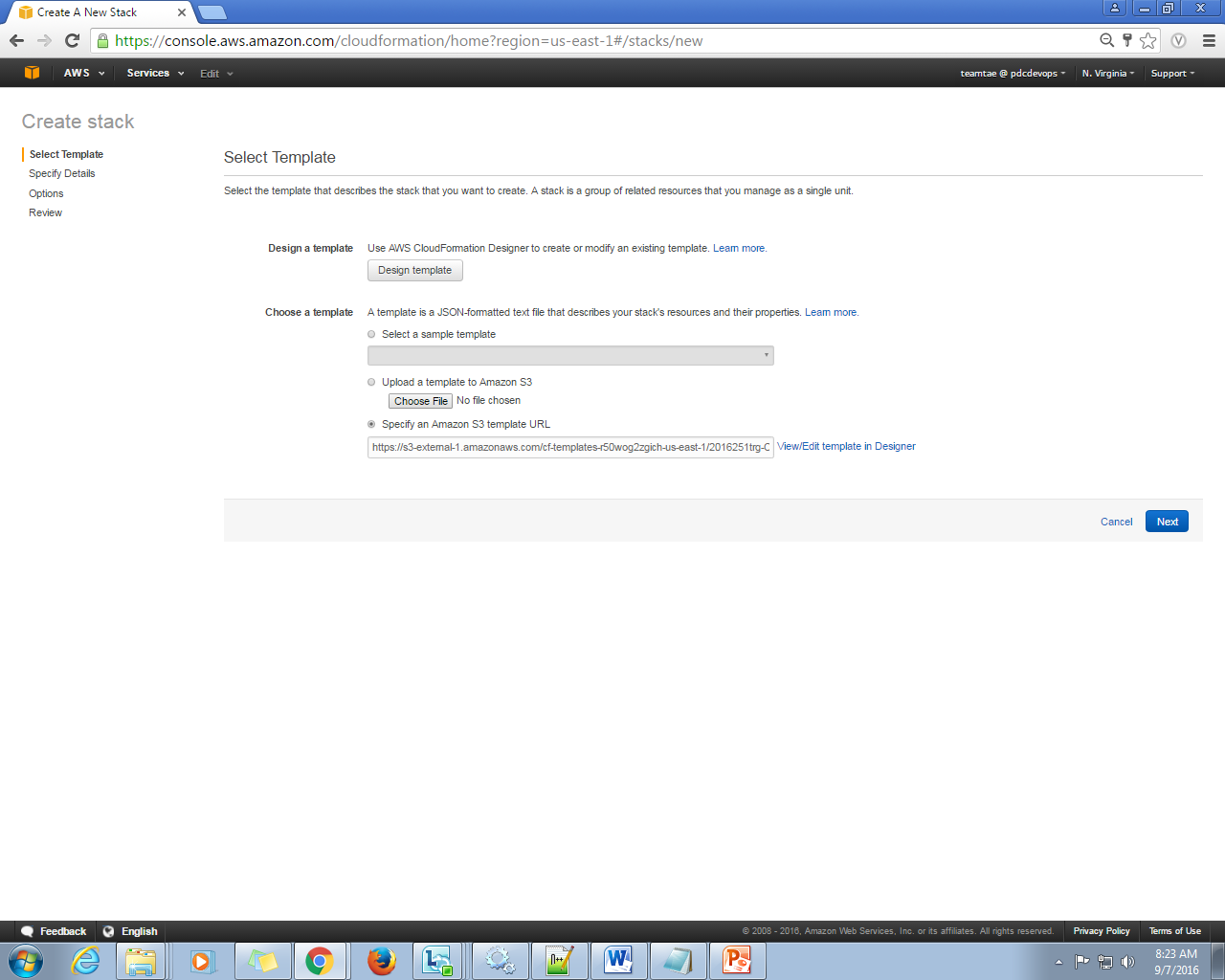
**2.4 WORKING THEORIES**

**2.4.1 Creating Stacks in AWS**

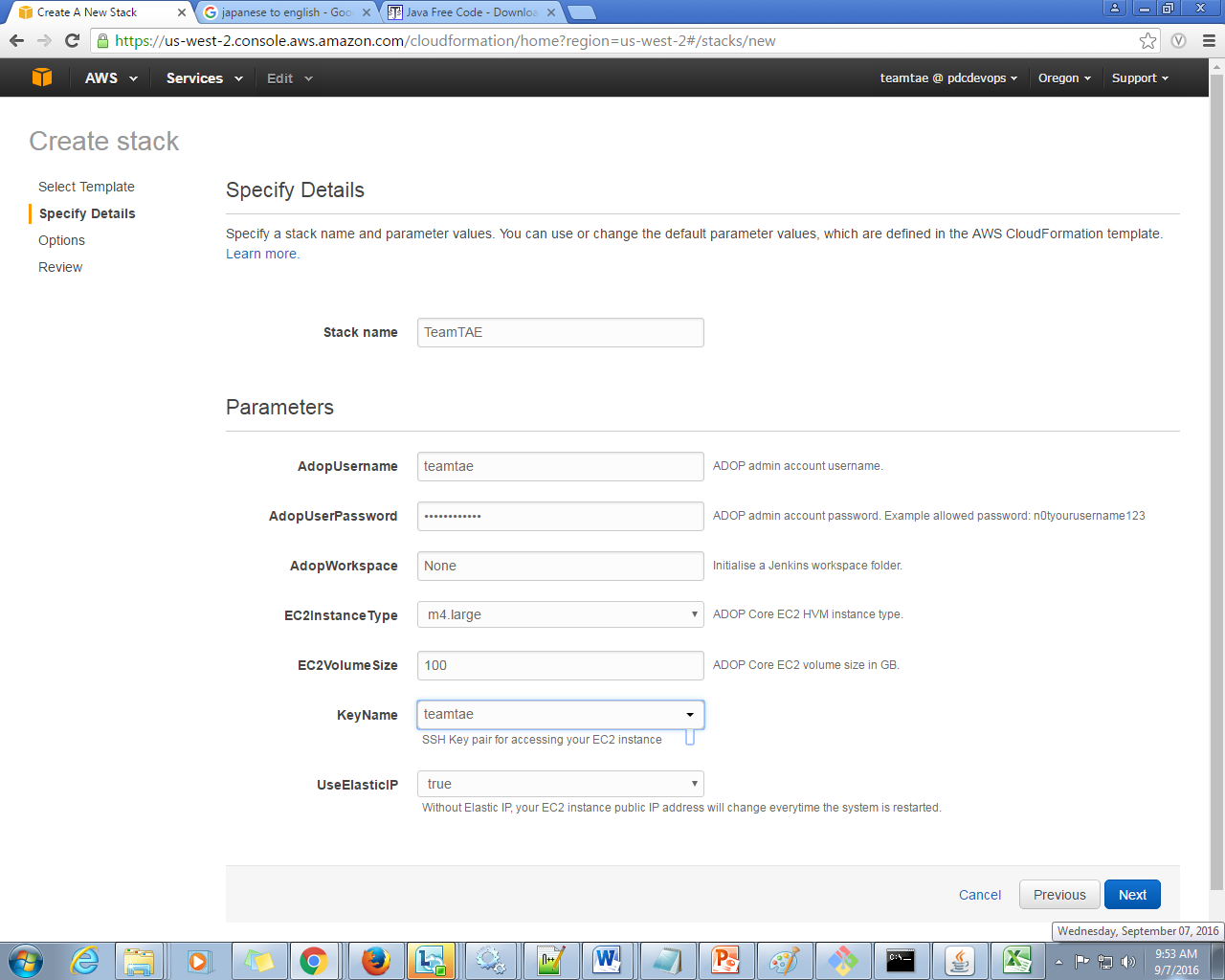
It represents a set of instances that will be managed for the entire system. Stack was named as TeamTae, instance type - m4.large, and instance size - 100GB.



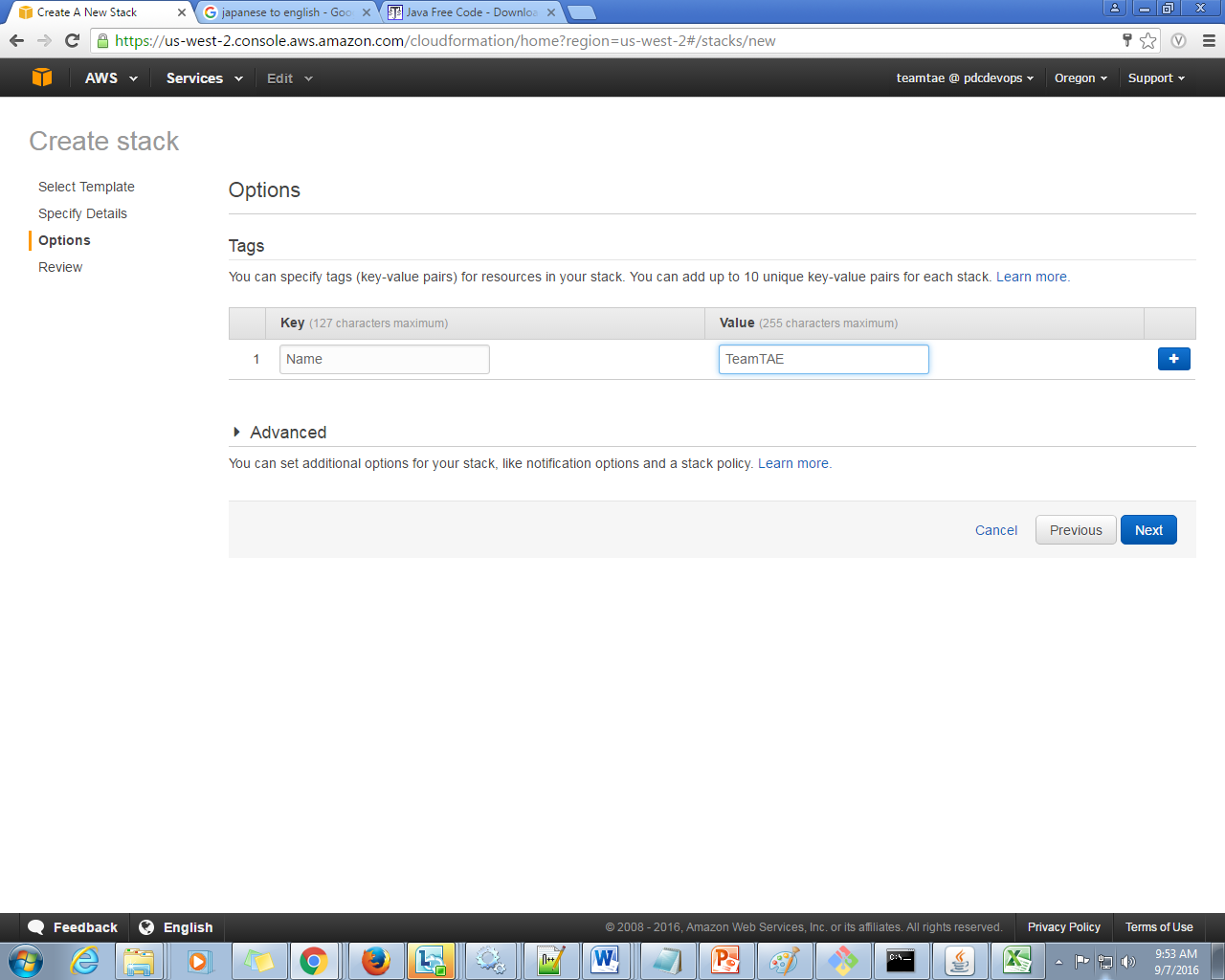
**Figure 2.3.** *Amazon Web Services home page*



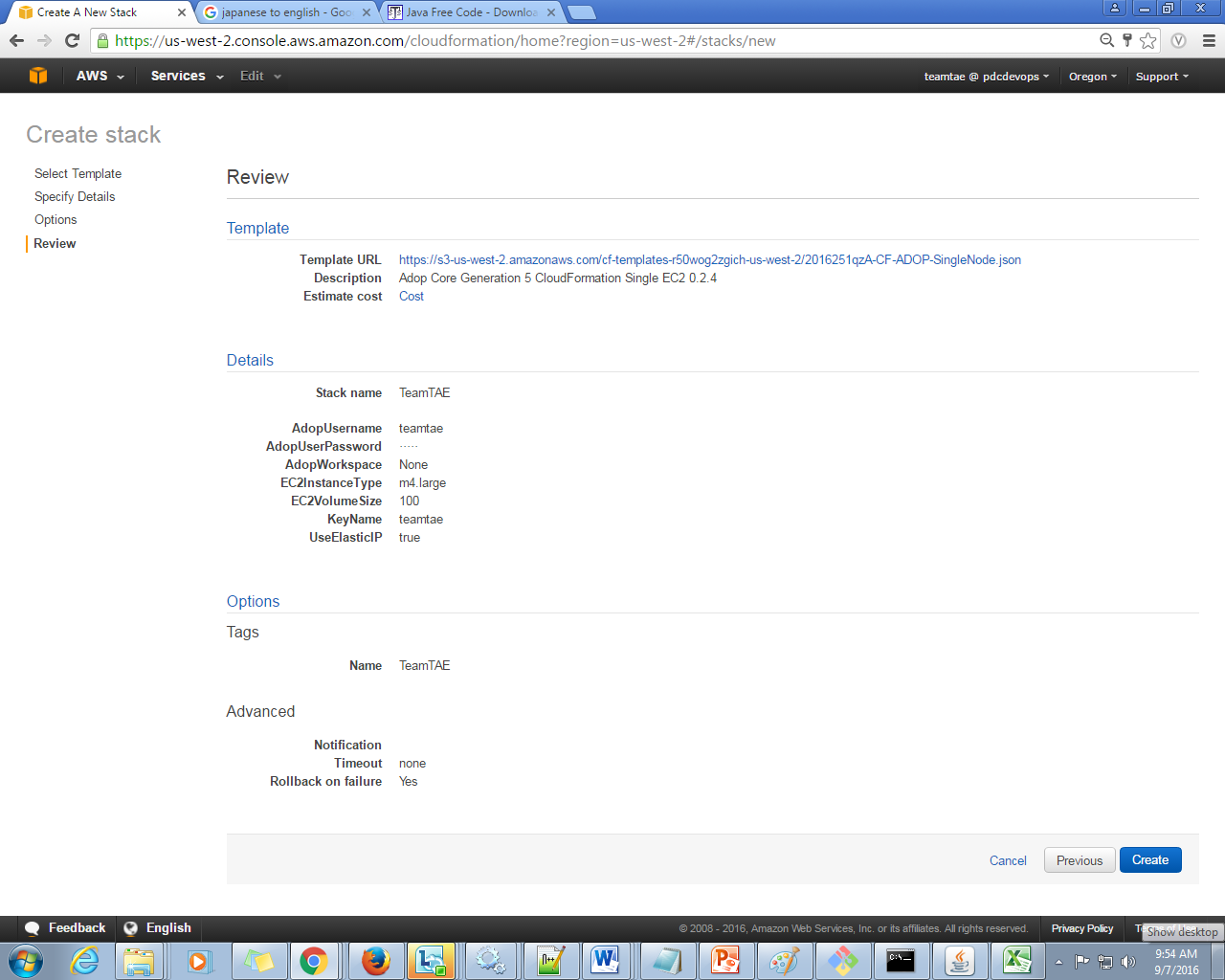
**Figure 2.4.** *Template URL*



**Figure 2.5.** *Details and Parameters*



**Figure 2.6.** *Tag*



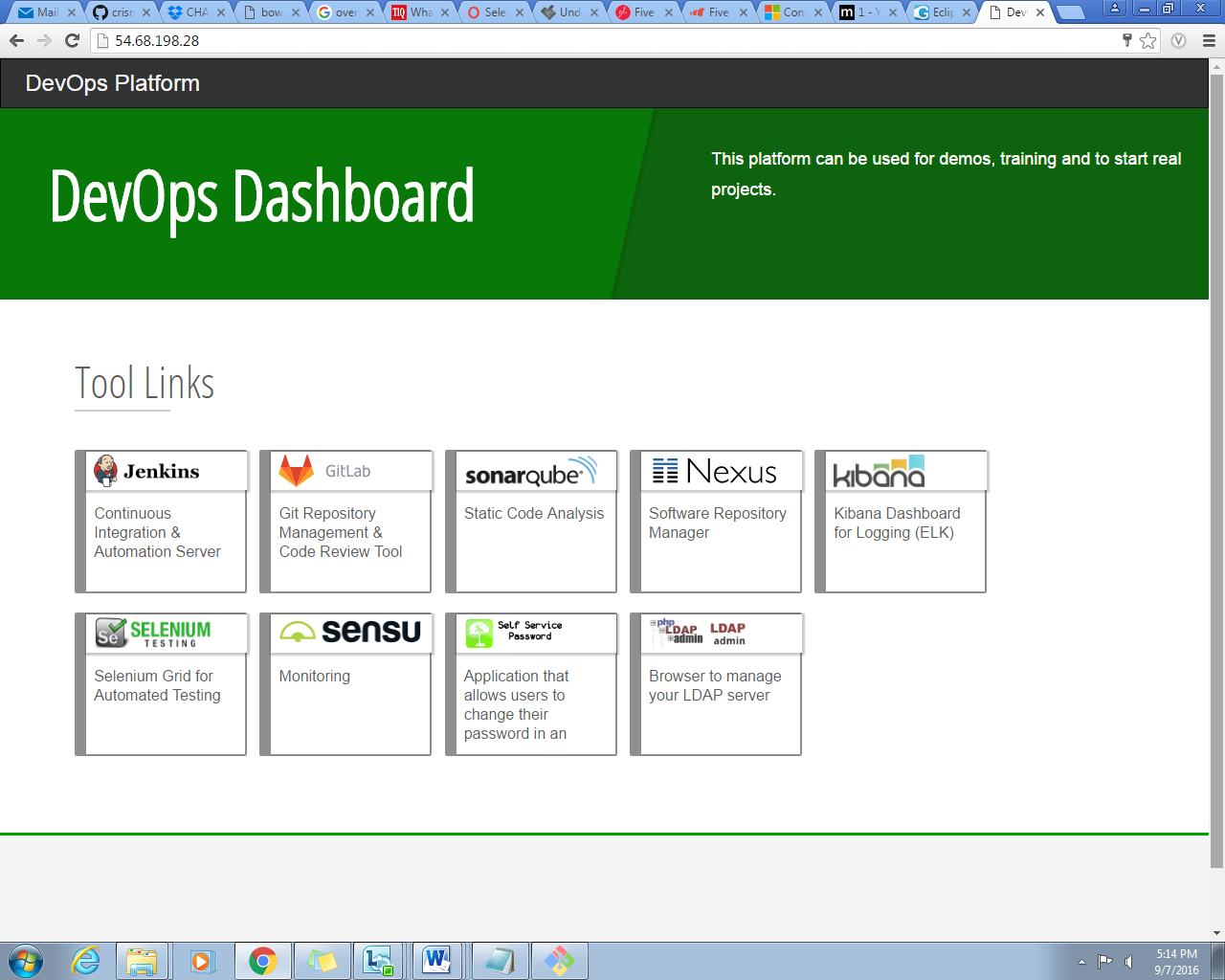
**Figure 2.7.** *Summary/Review*



**Figure 2.8.** *Stack Status*

**2.4.2 DevOps Platform**

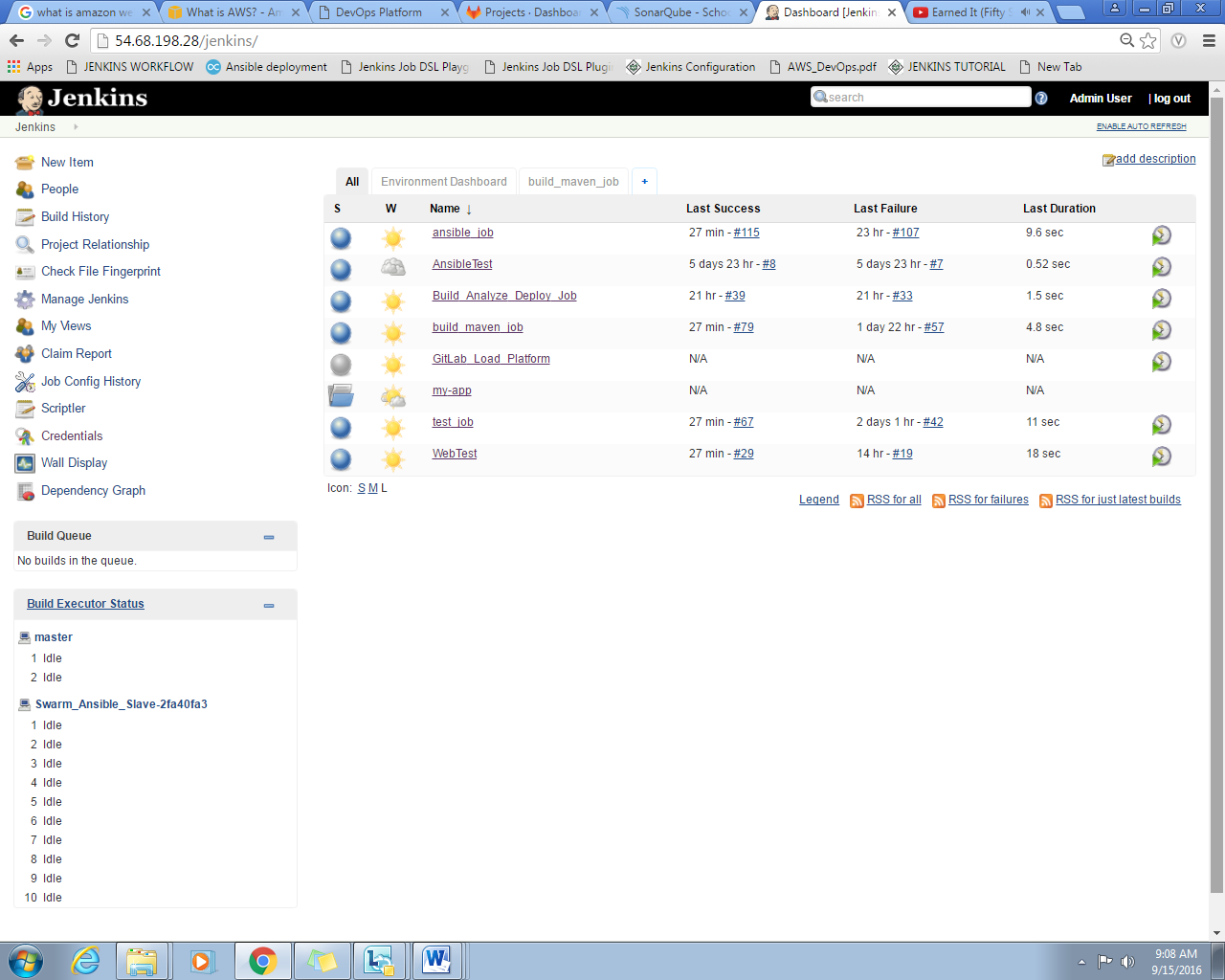
DevOps platform showcase the tools that will be used in the system like Jenkins, GitLab, Sonarqube, and Selenium. In accessing the platform, <http://54.68.198.28/> was used.



**Figure 2.9.** *DevOps Dashboard*

**2.4.3. Jenkins**

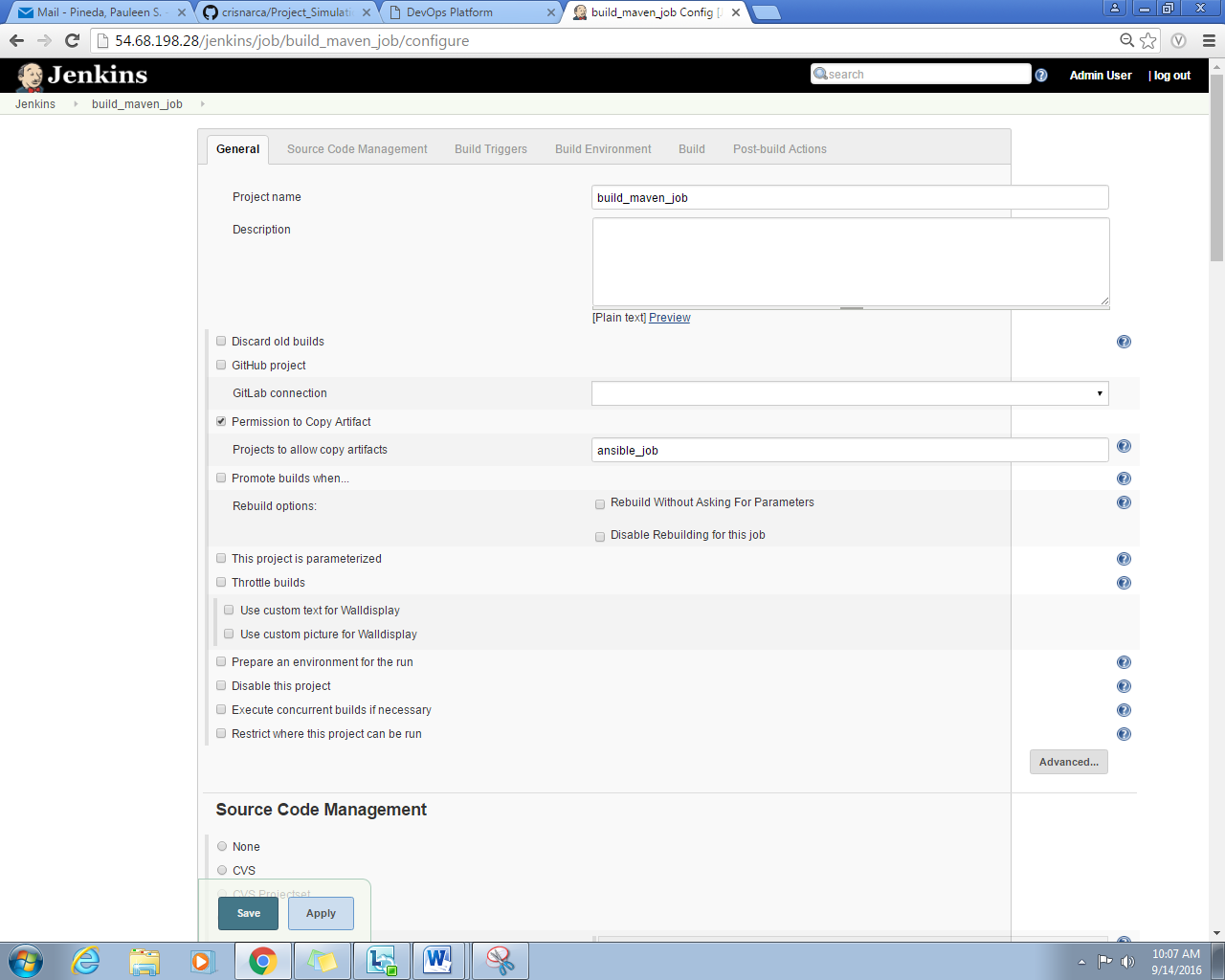
Different jobs for different tools are listed in the Jenkins dashboard. It shows the current name, status, last success/failure/duration of the job created. Blue indicates successful job, while red when it is not successful.



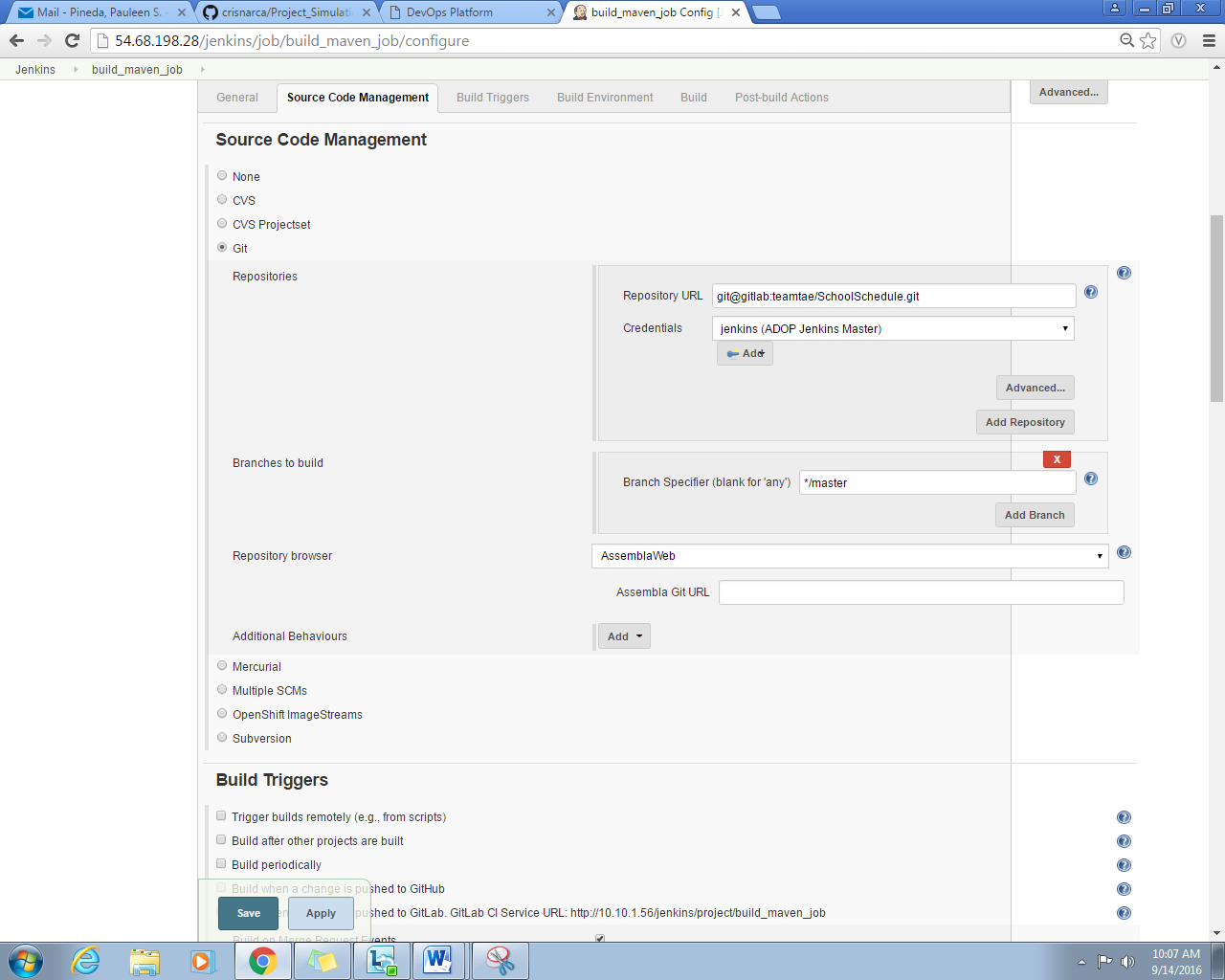
**Figure 2.10.** *Jenkins dashboard*

**2.4.4 Maven Configuration**

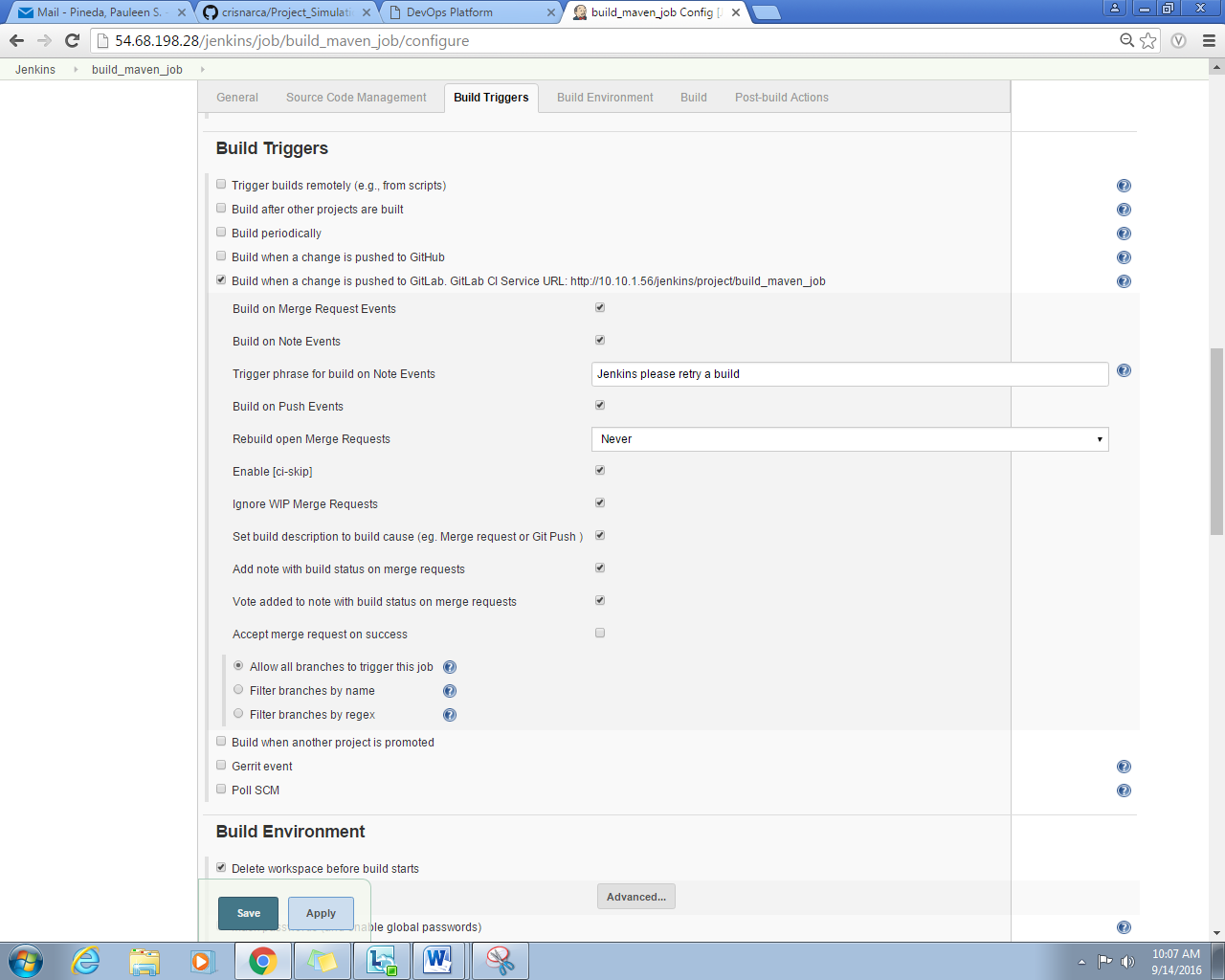
In the build\_maven\_job, ……



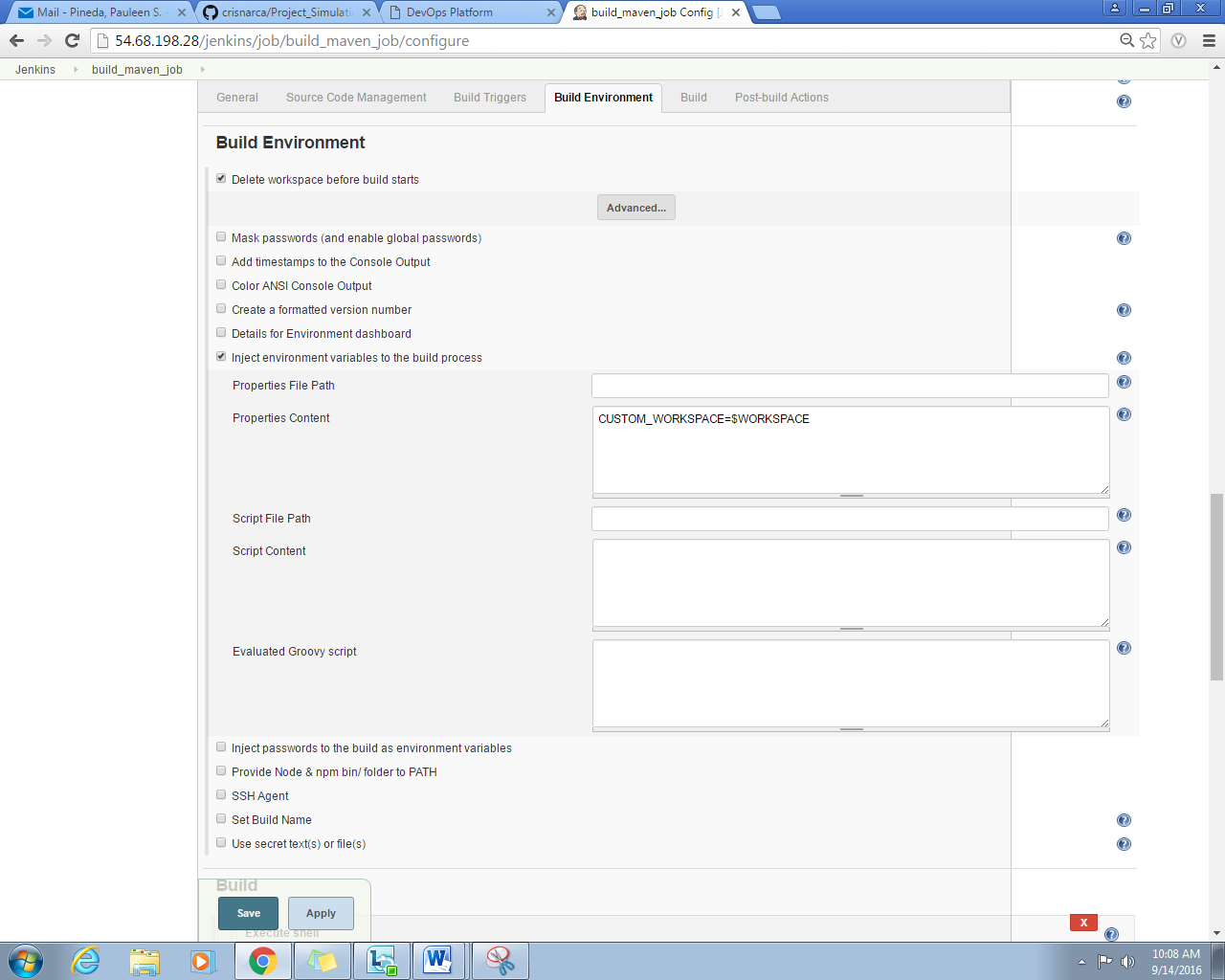
**Figure 2.11.** *General Configuration*



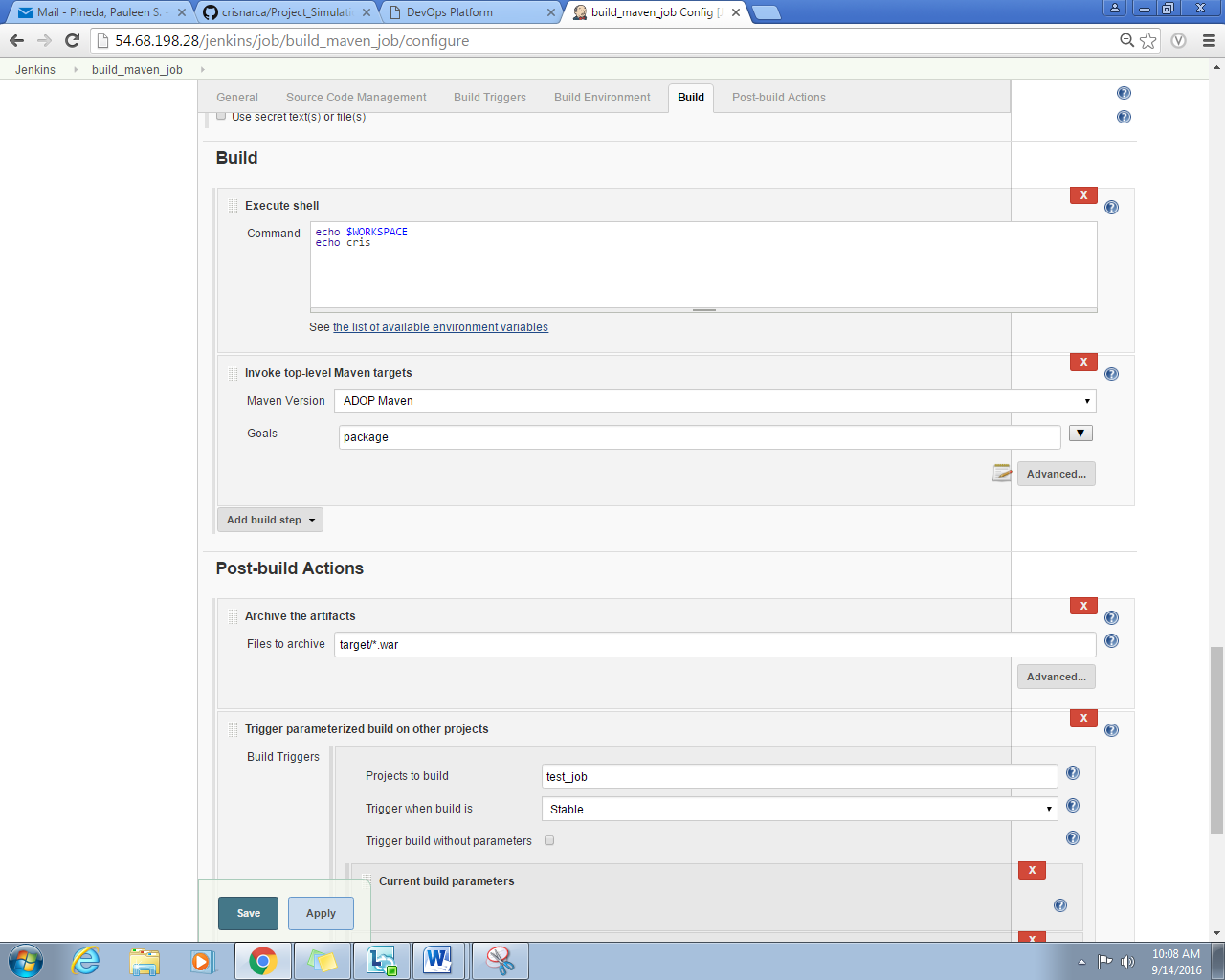
**Figure 2.12.** *Source Code Management Configuration*



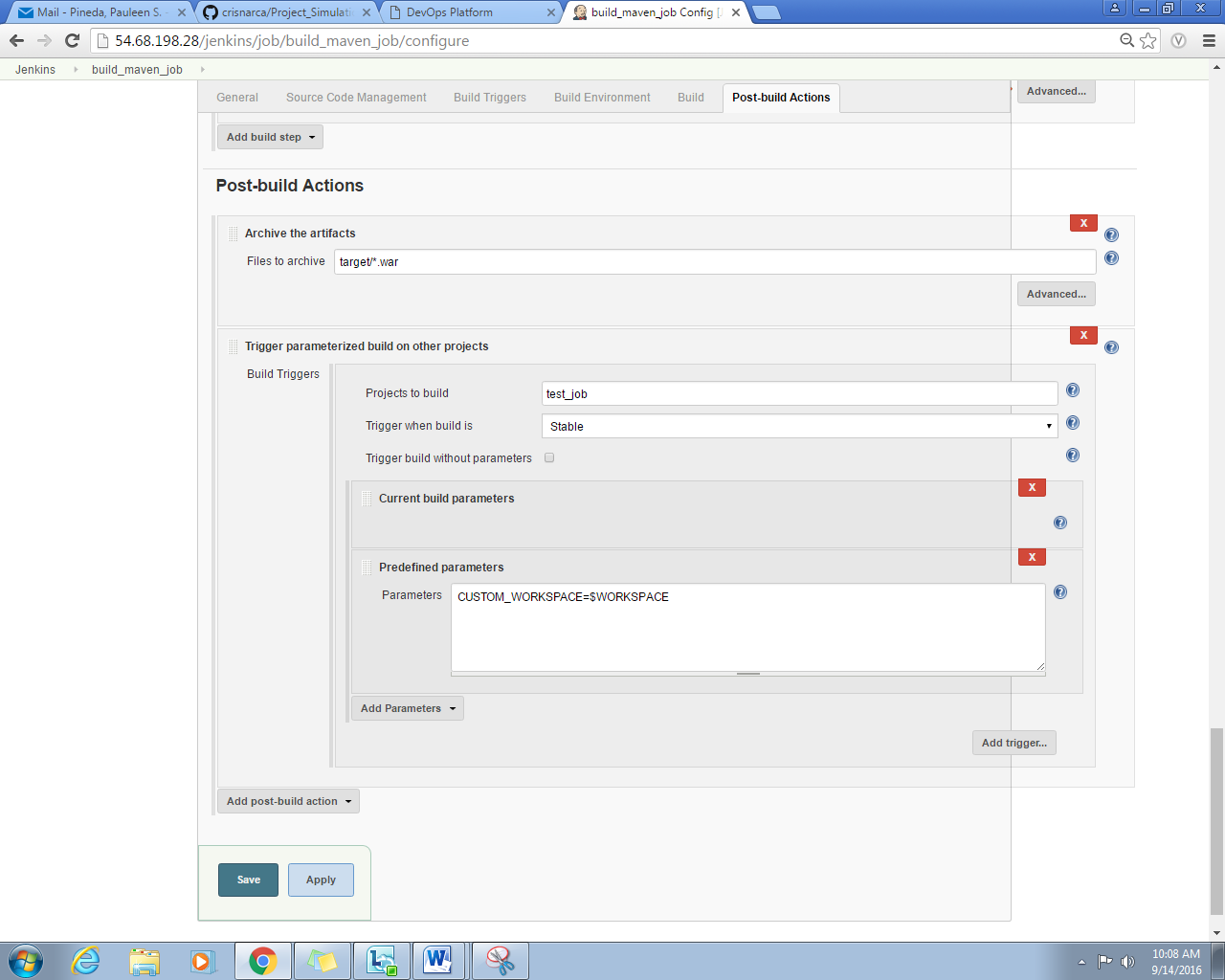
**Figure 2.13.** *Build Trigger Configuration*



**Figure 2.14.** *Build Environment Configuration*

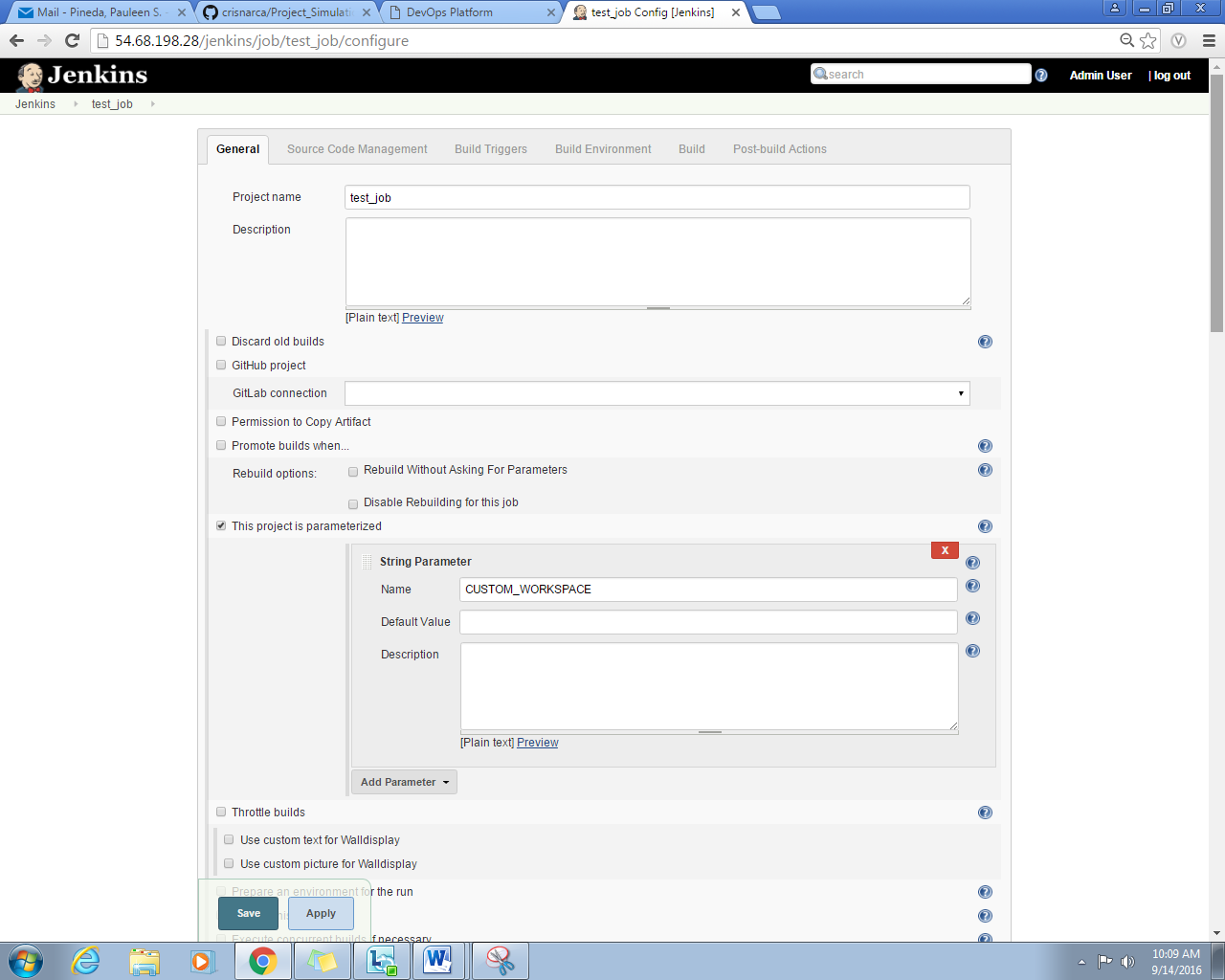


**Figure 2.15.** *Build Configuration*

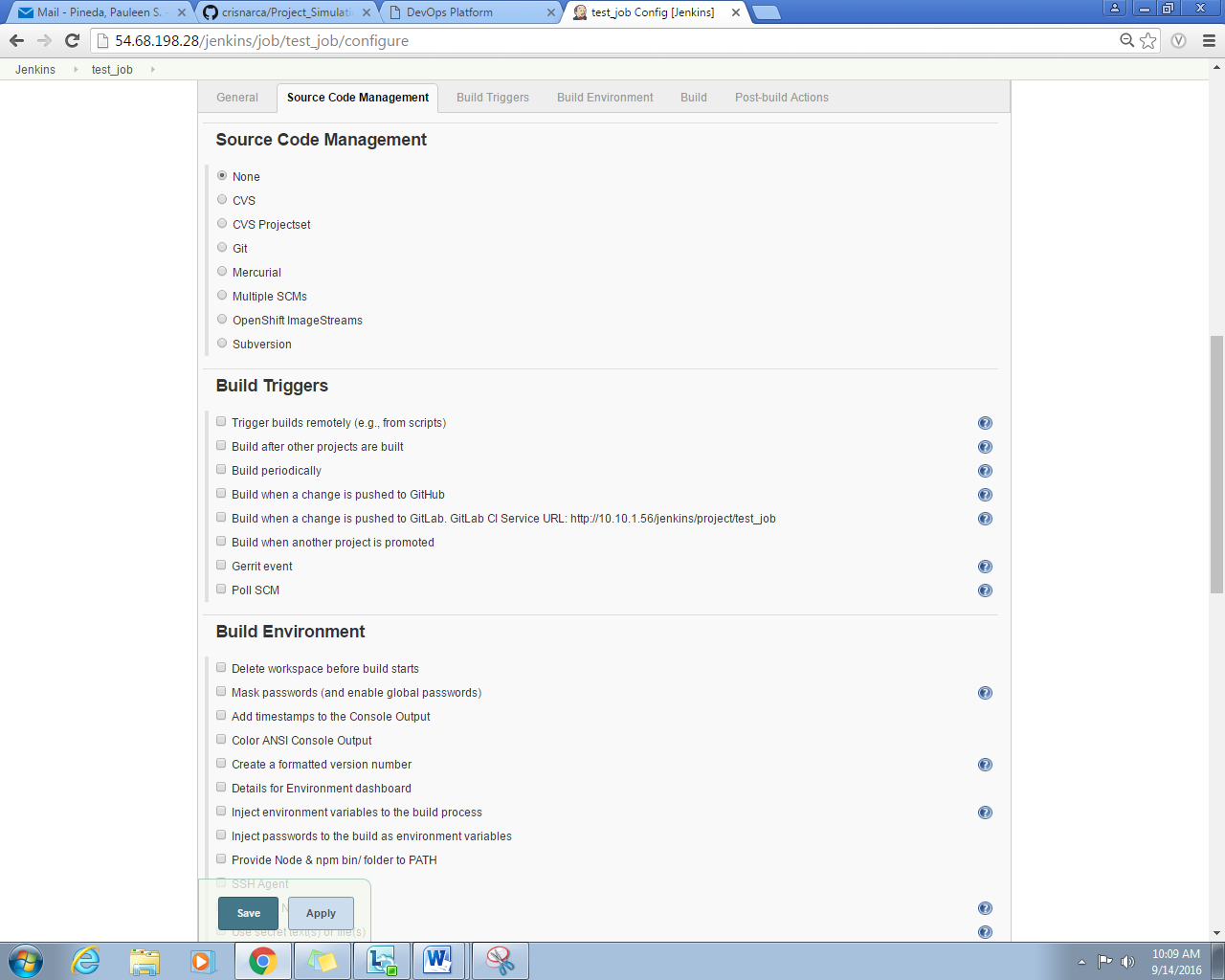


**Figure 2.16.** *Post-Build Actions Configuration*

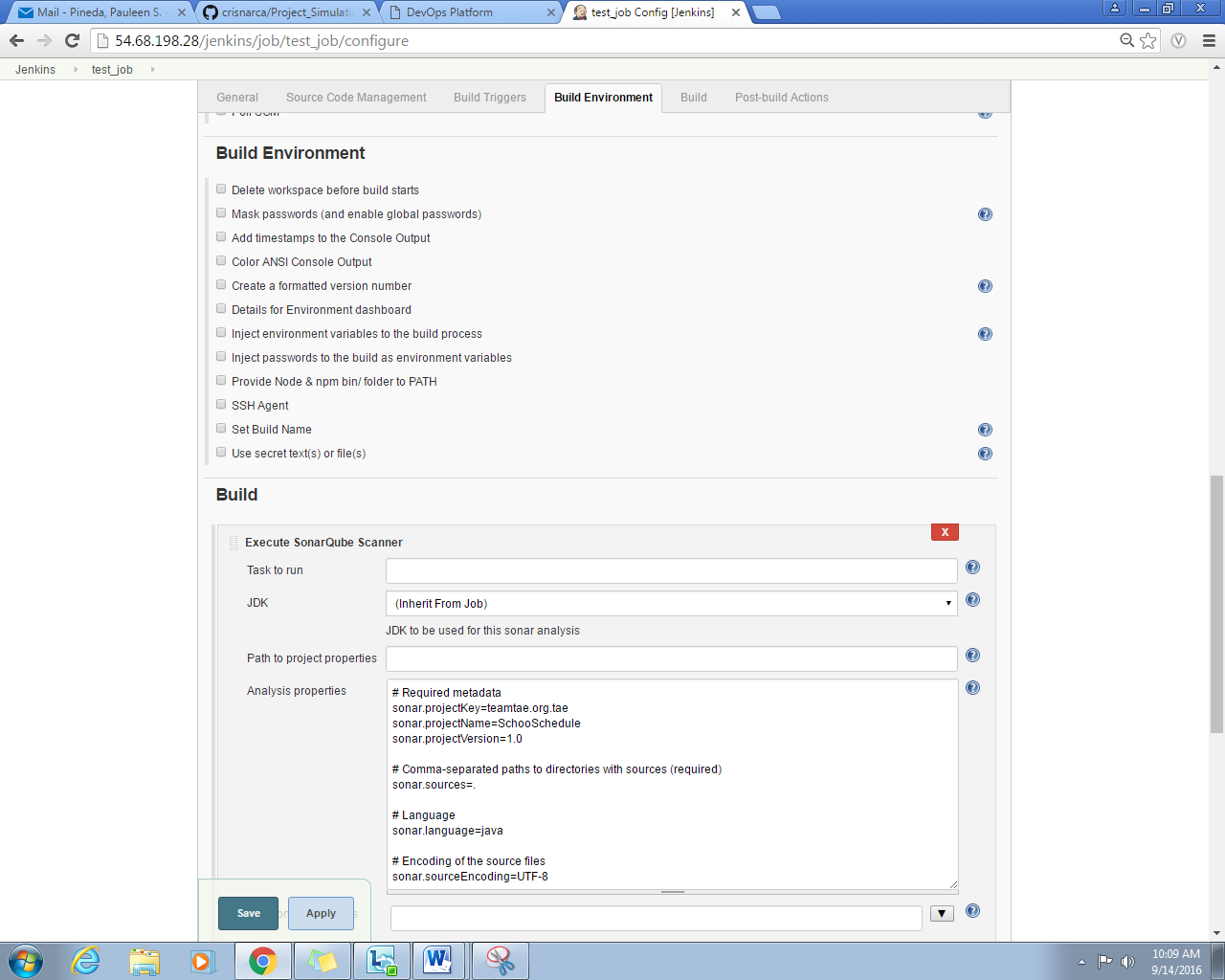
**2.4.5 Sonarqube Configuration**



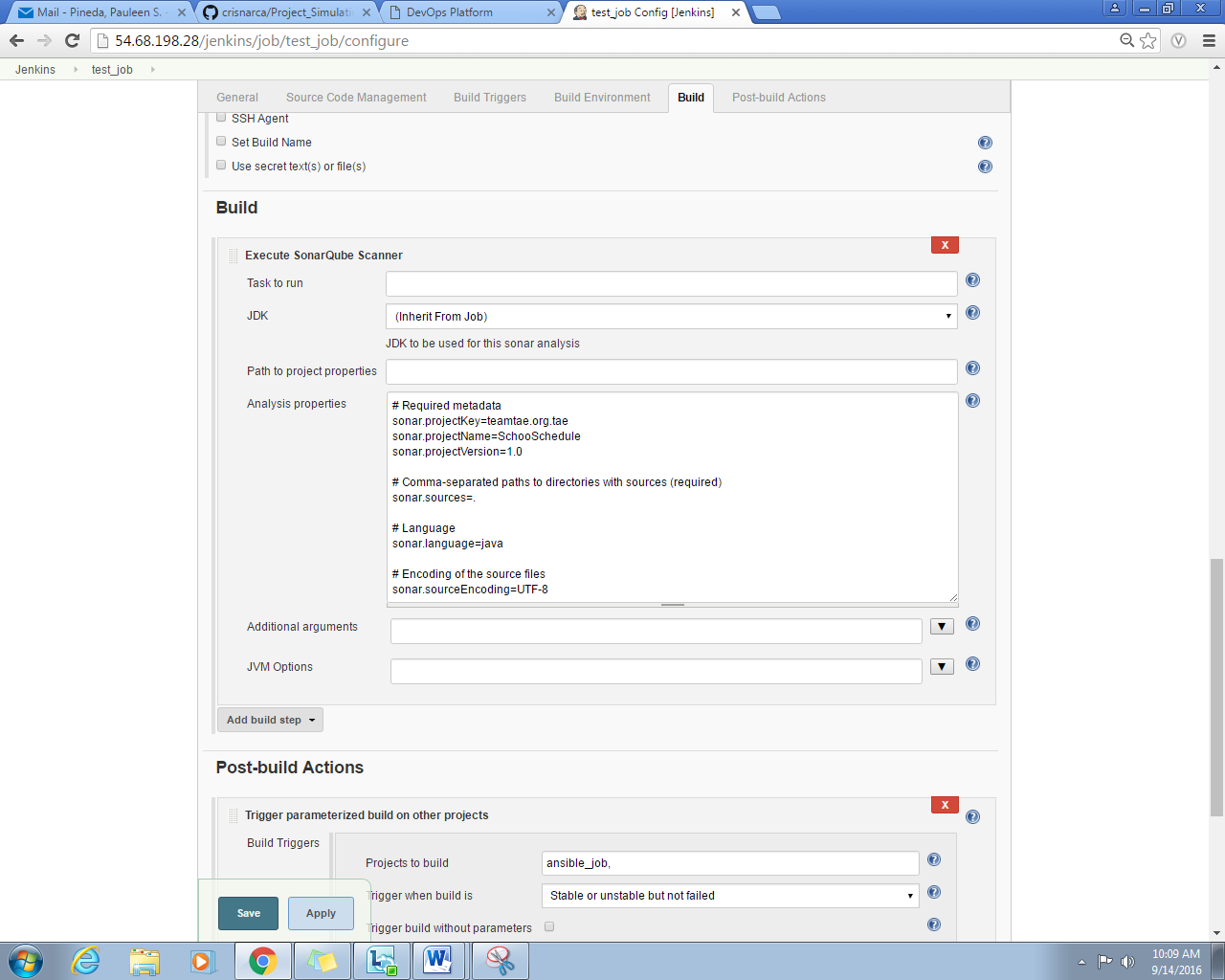
**Figure 2.17.** *General Configuration*



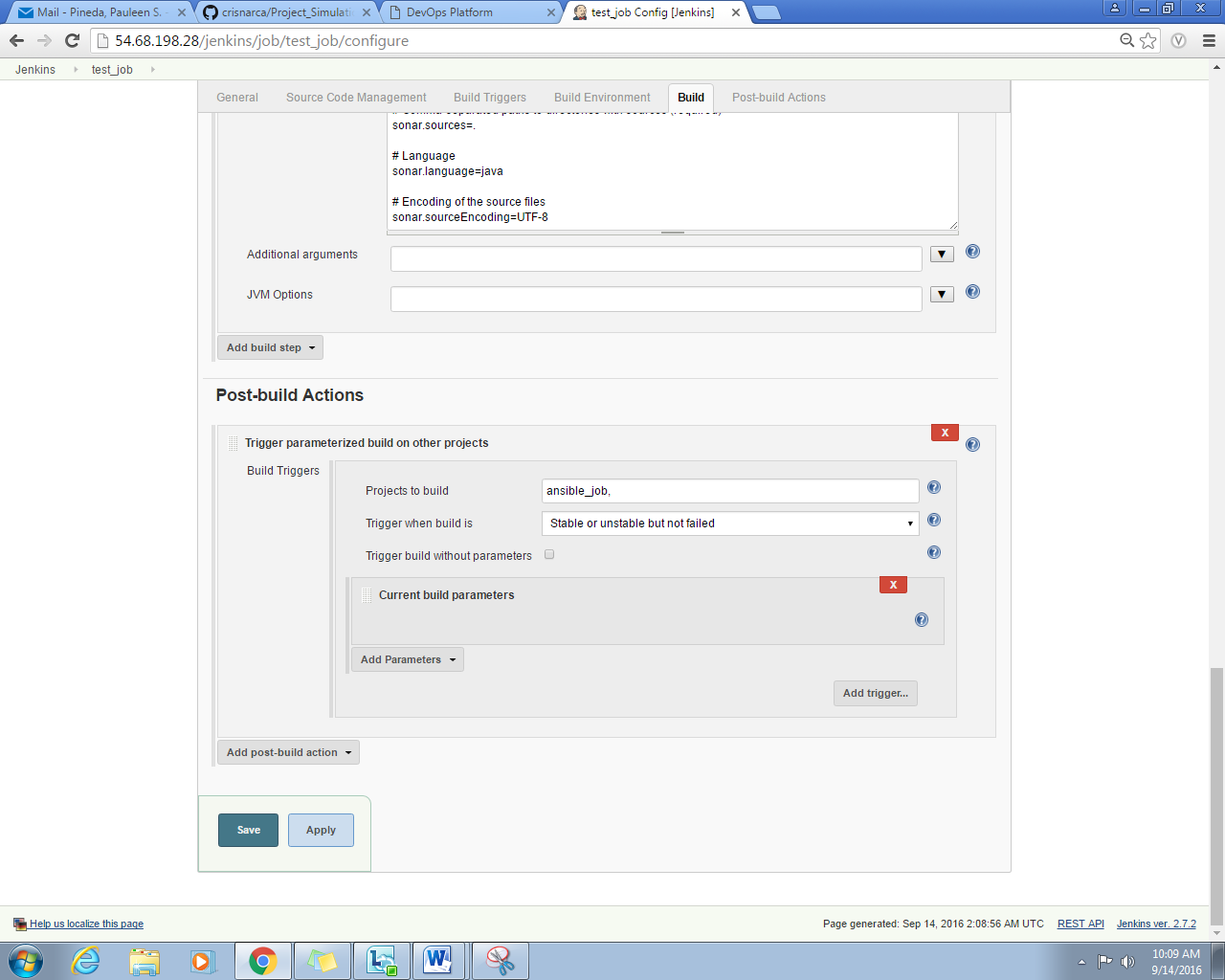
**Figure 2.18.** *Source Code Management Configuration*



**Figure 2.19.** *Build Environment Configuration*

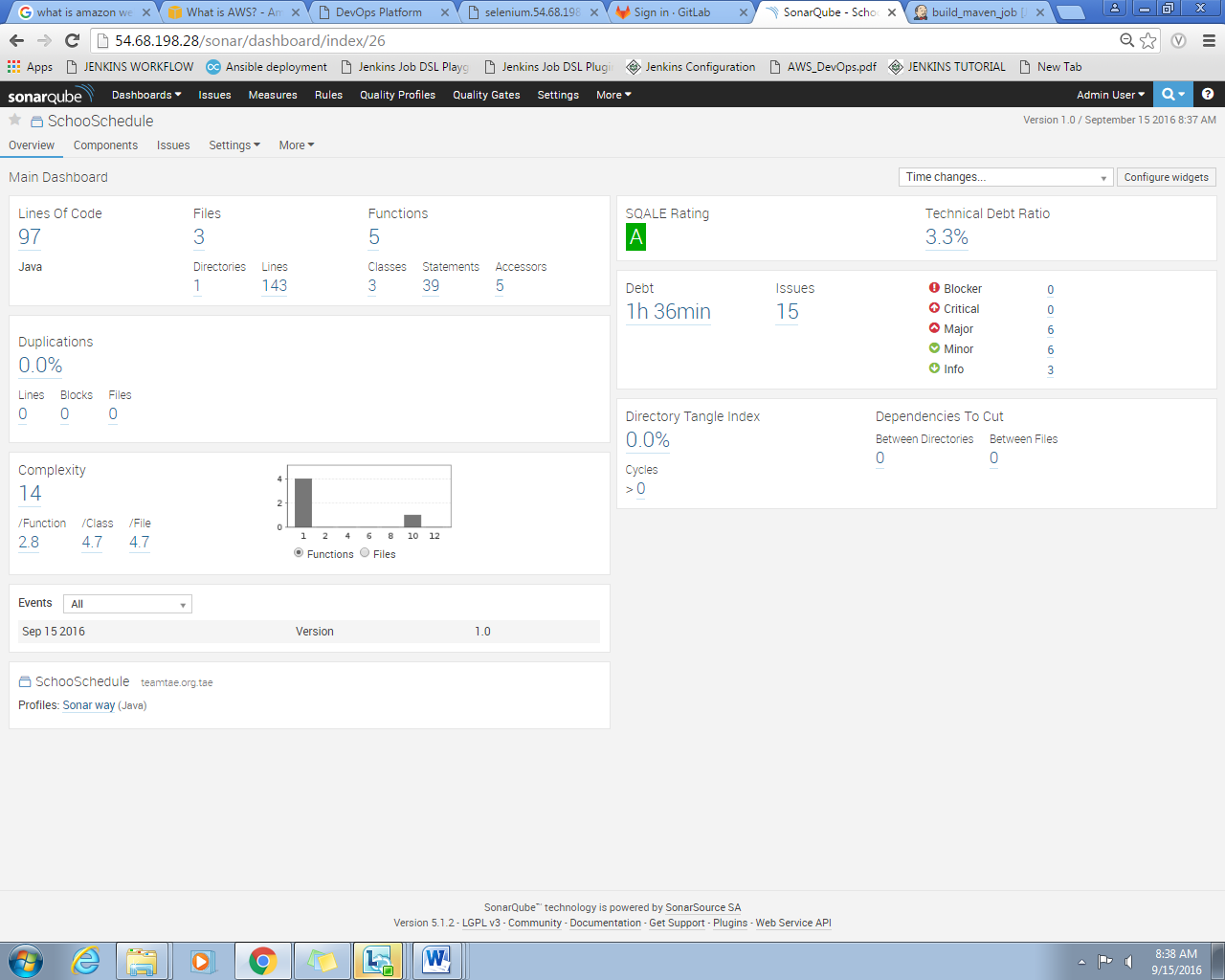


**Figure 2.20.** *Build Configuration*



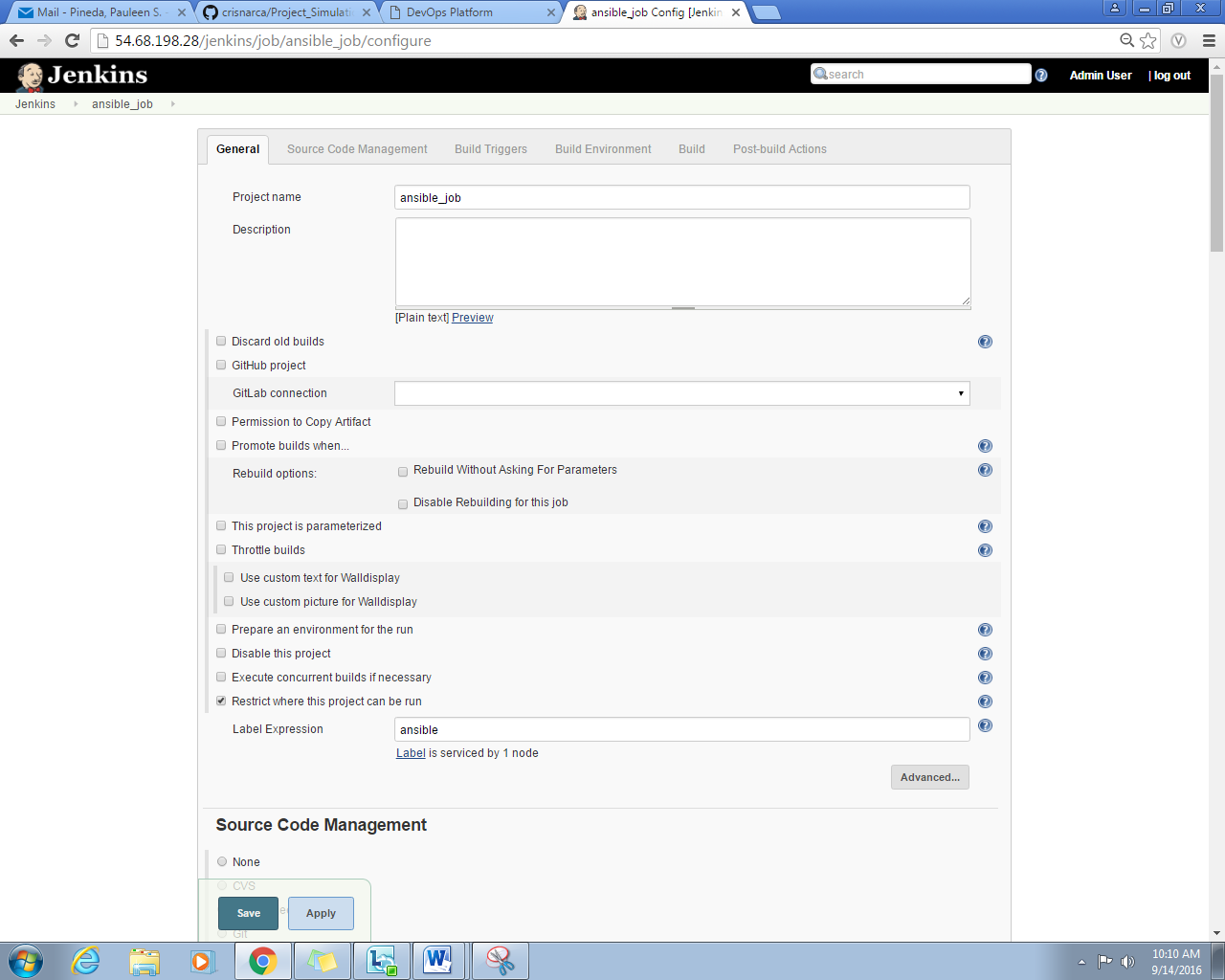
**Figure 2.21.** *Post-Build Action Configuration*

In the main dashboard of sonarqube, details of the code like complexity, functions the number of linesof codes, and category of issues are indicated. Summary of the quality of codes are shown.

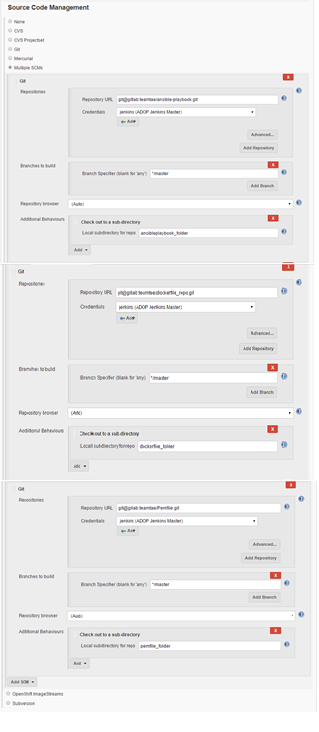


**Figure 2.22.** *Sonarqube homepage*

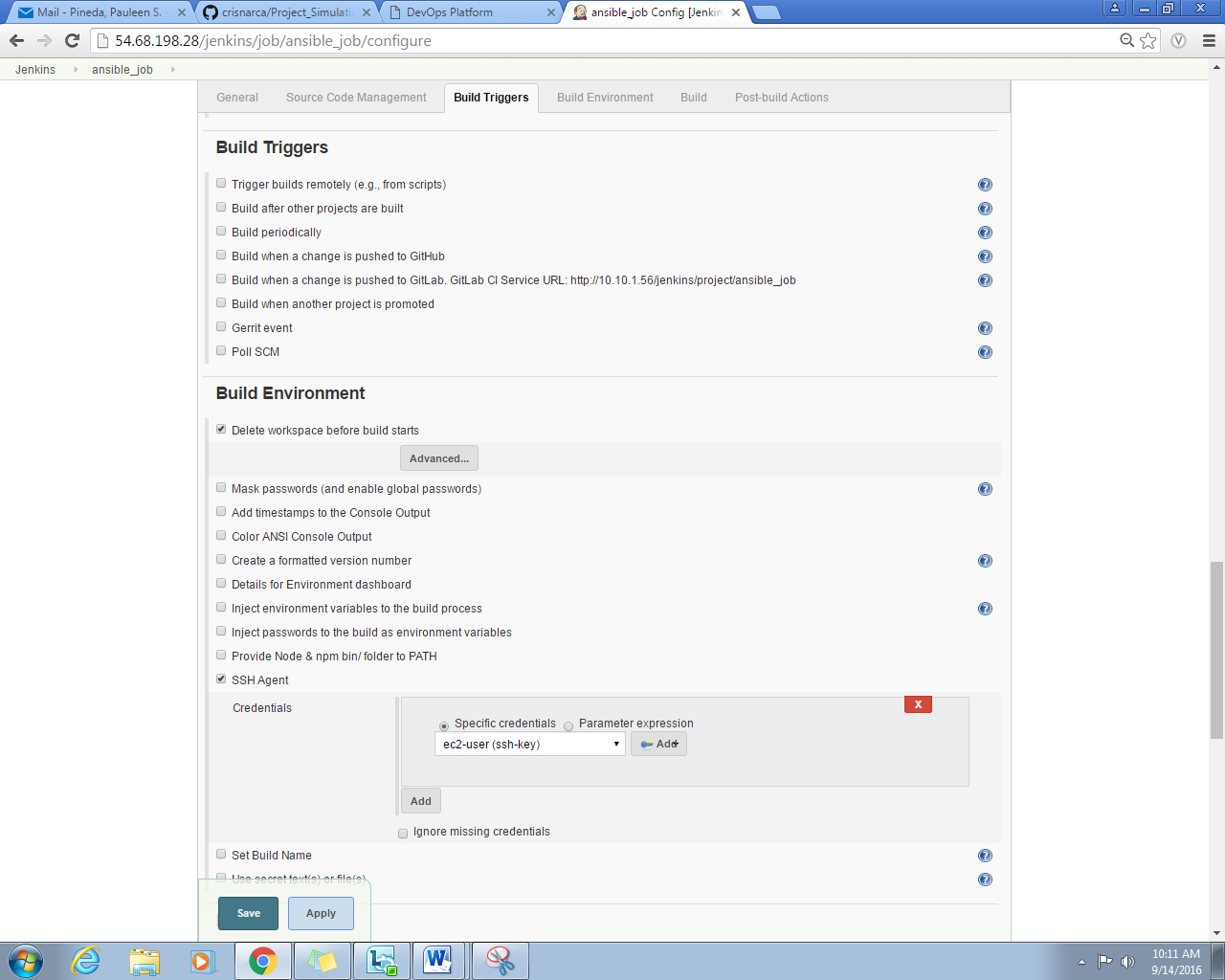
**2.4.6 Ansible Configuration**



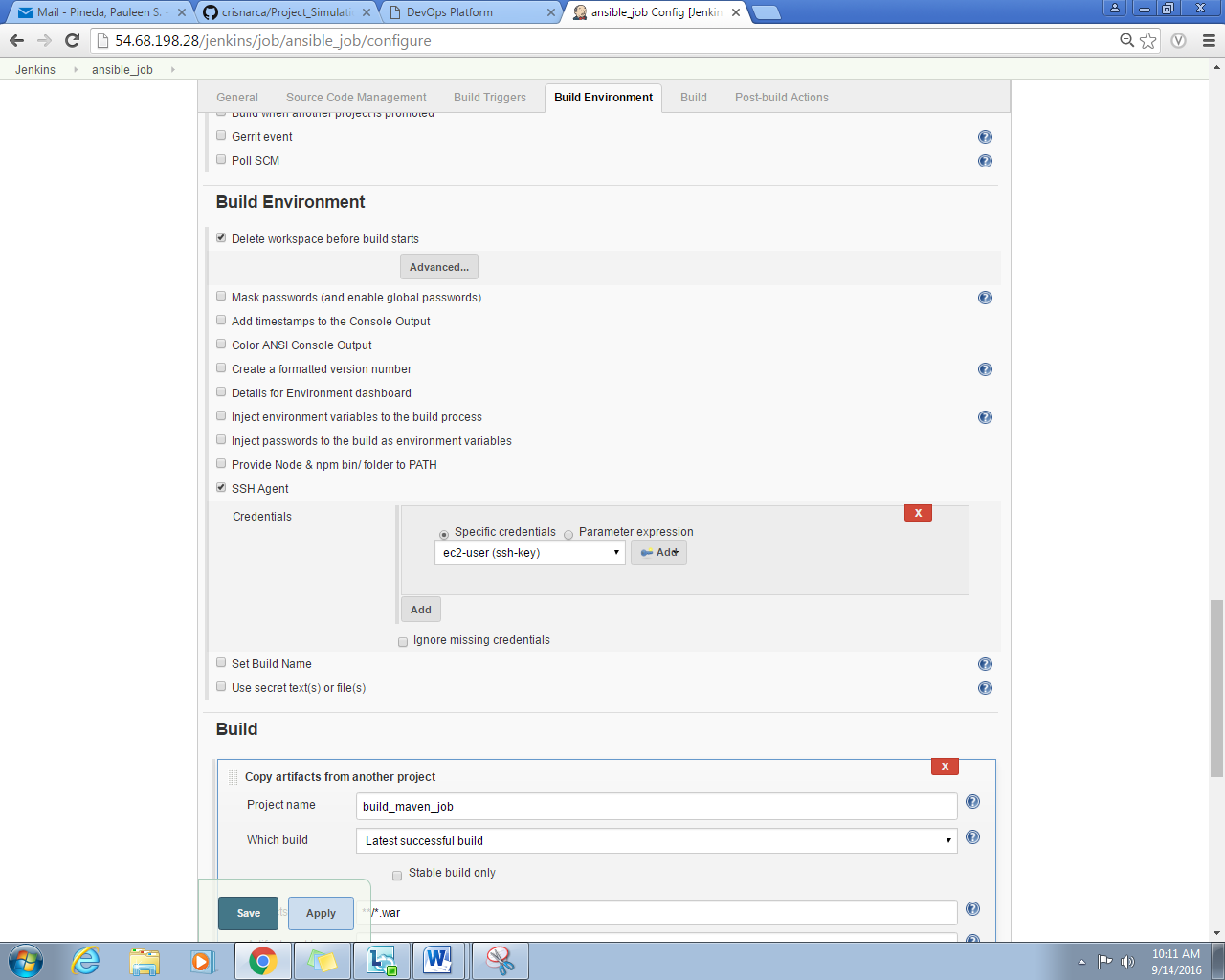
**Figure 2.23.** *General Configuration*



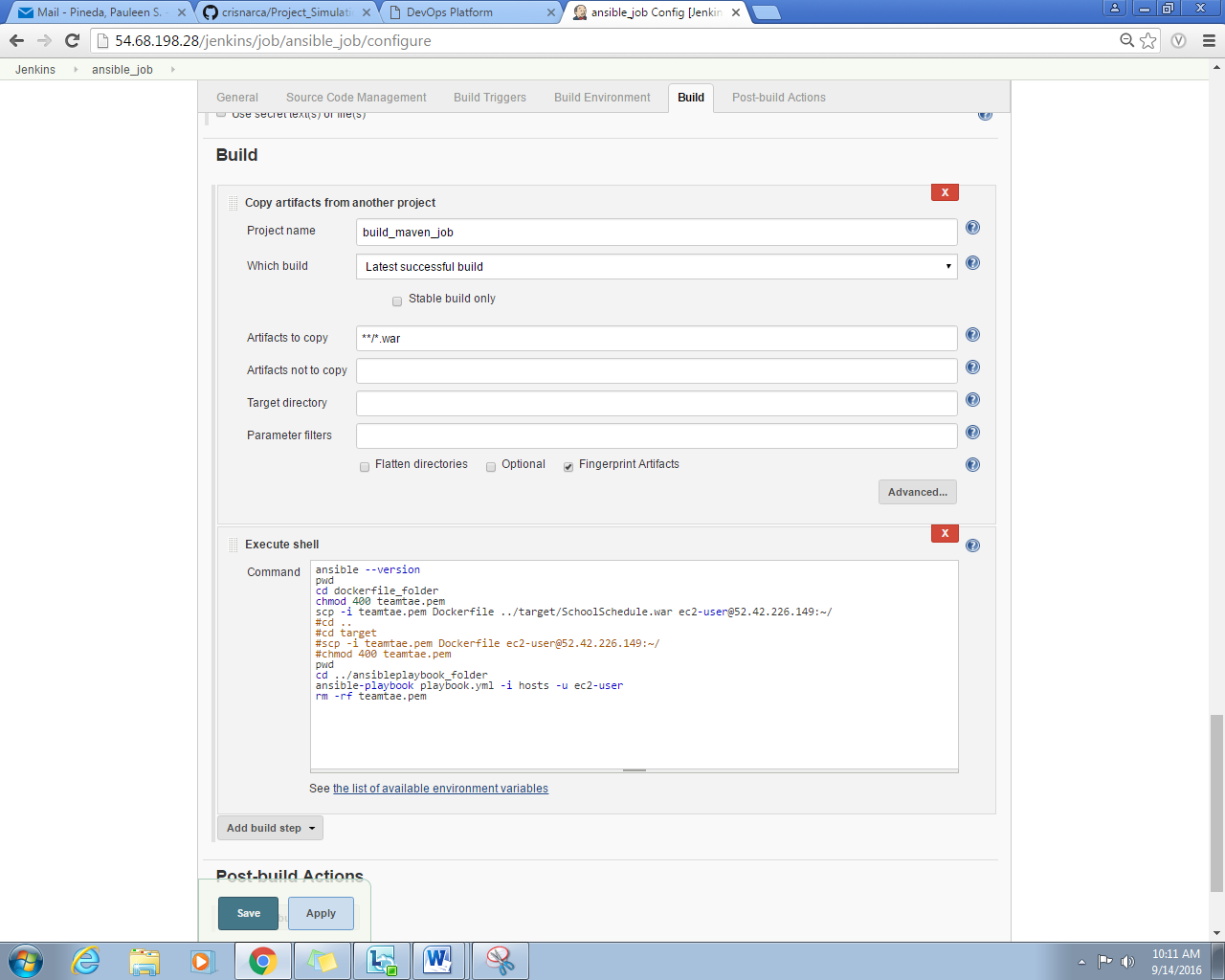
**Figure 2.24.** *Source Code Management Configuration*



**Figure 2.25.** *Build Trigger Configuration*

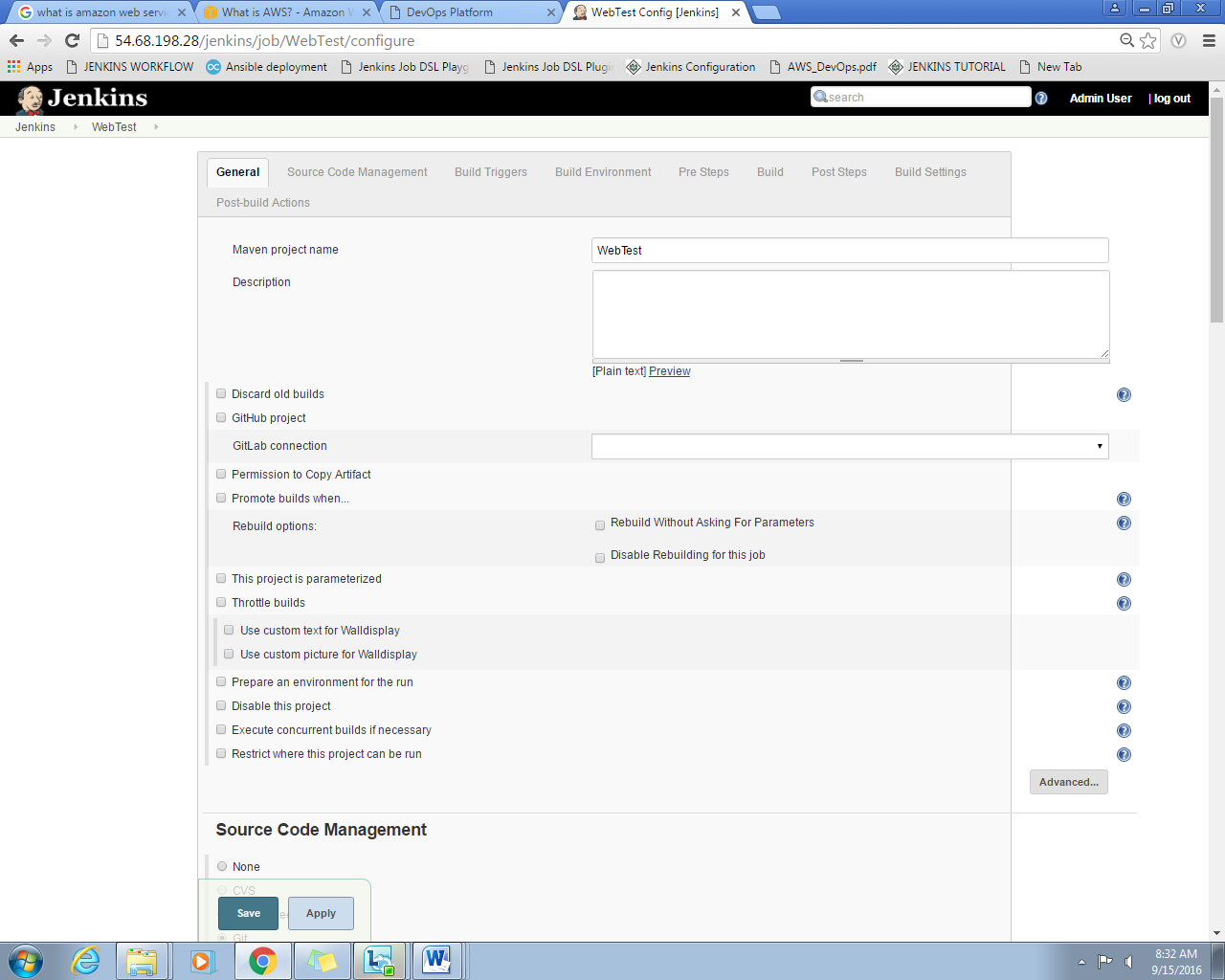


**Figure 2.26.** *Build Environment Configuration*

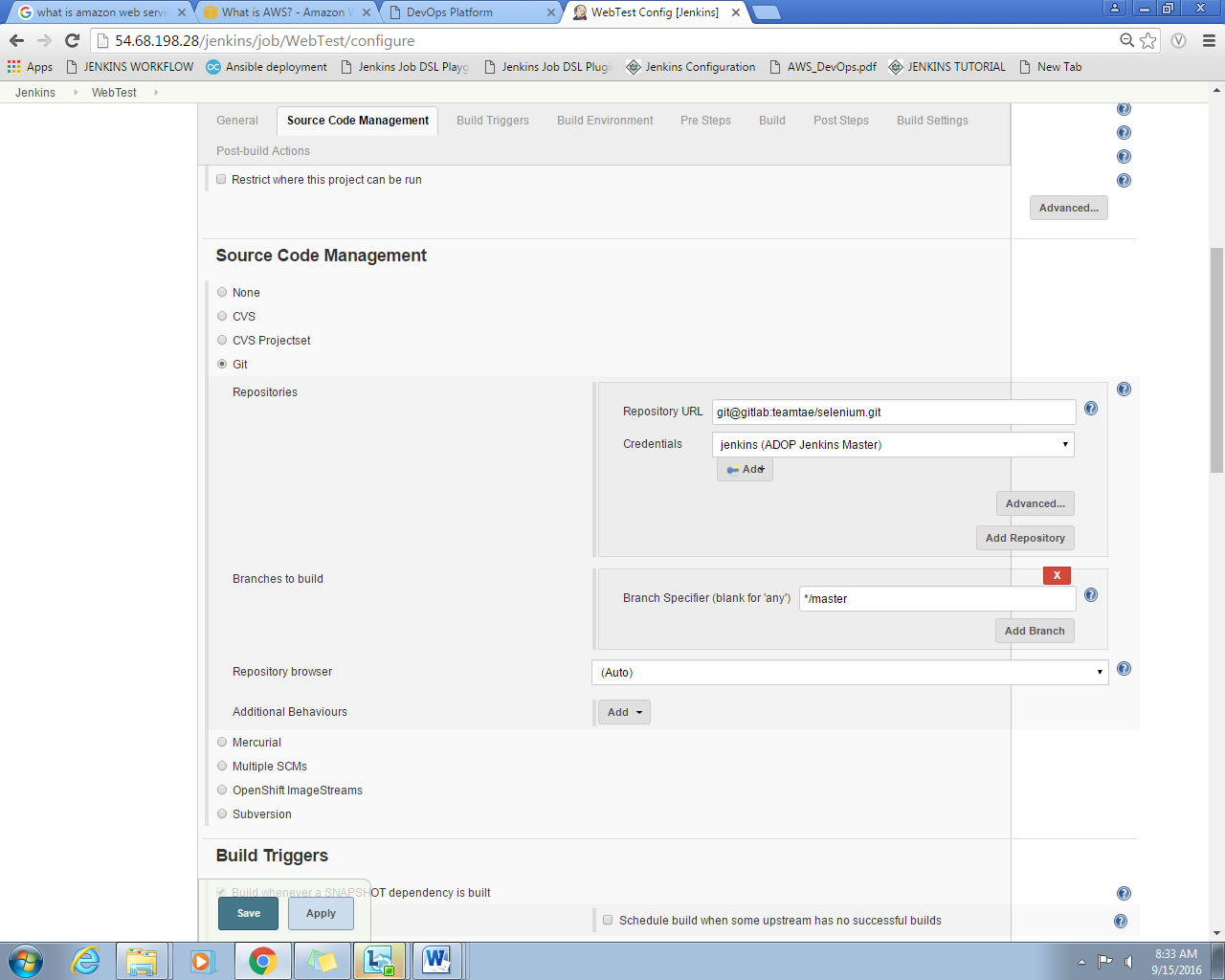


**Figure 2.27.** *Build Configuration*

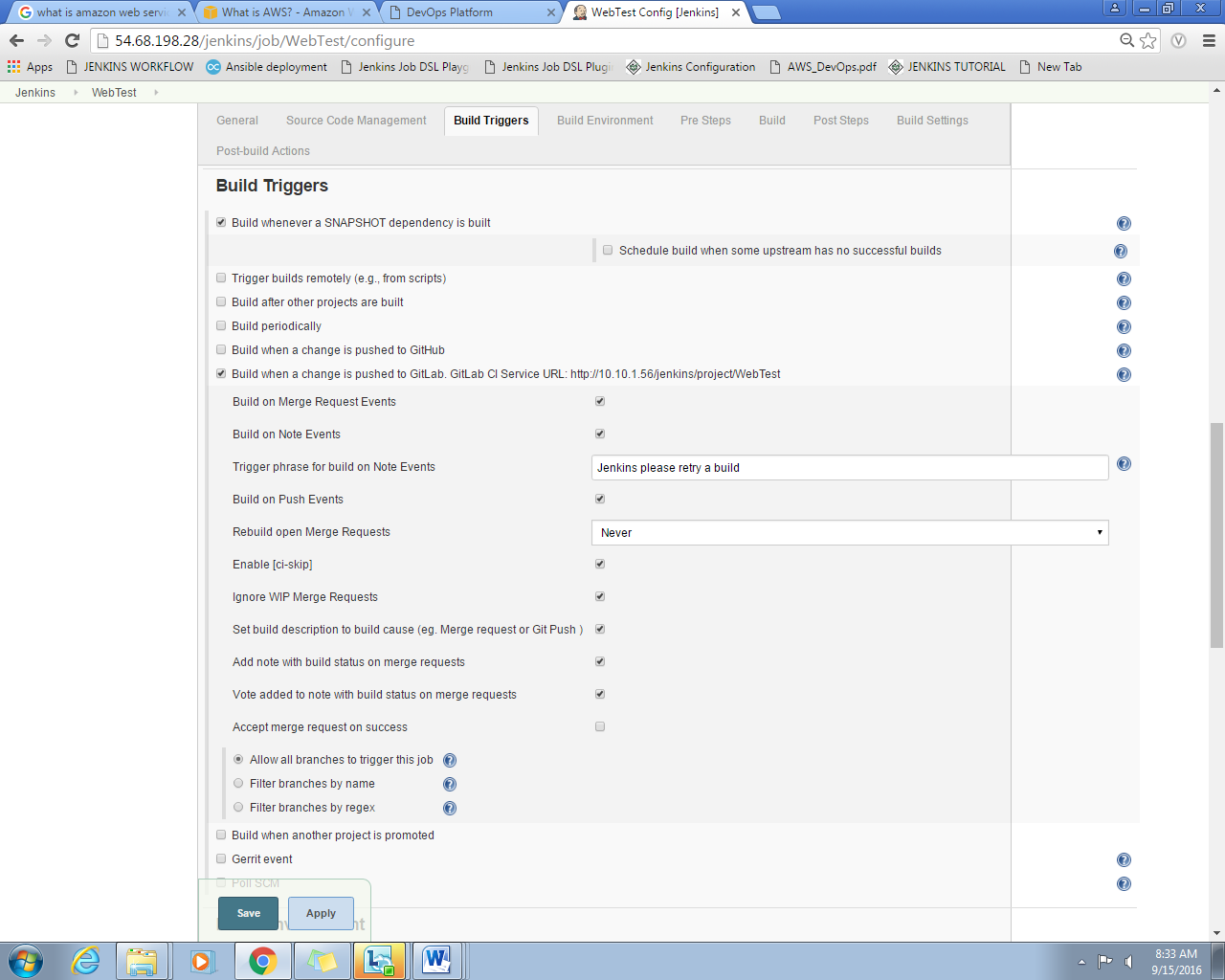
**2.4.7 Selenium Configuration**



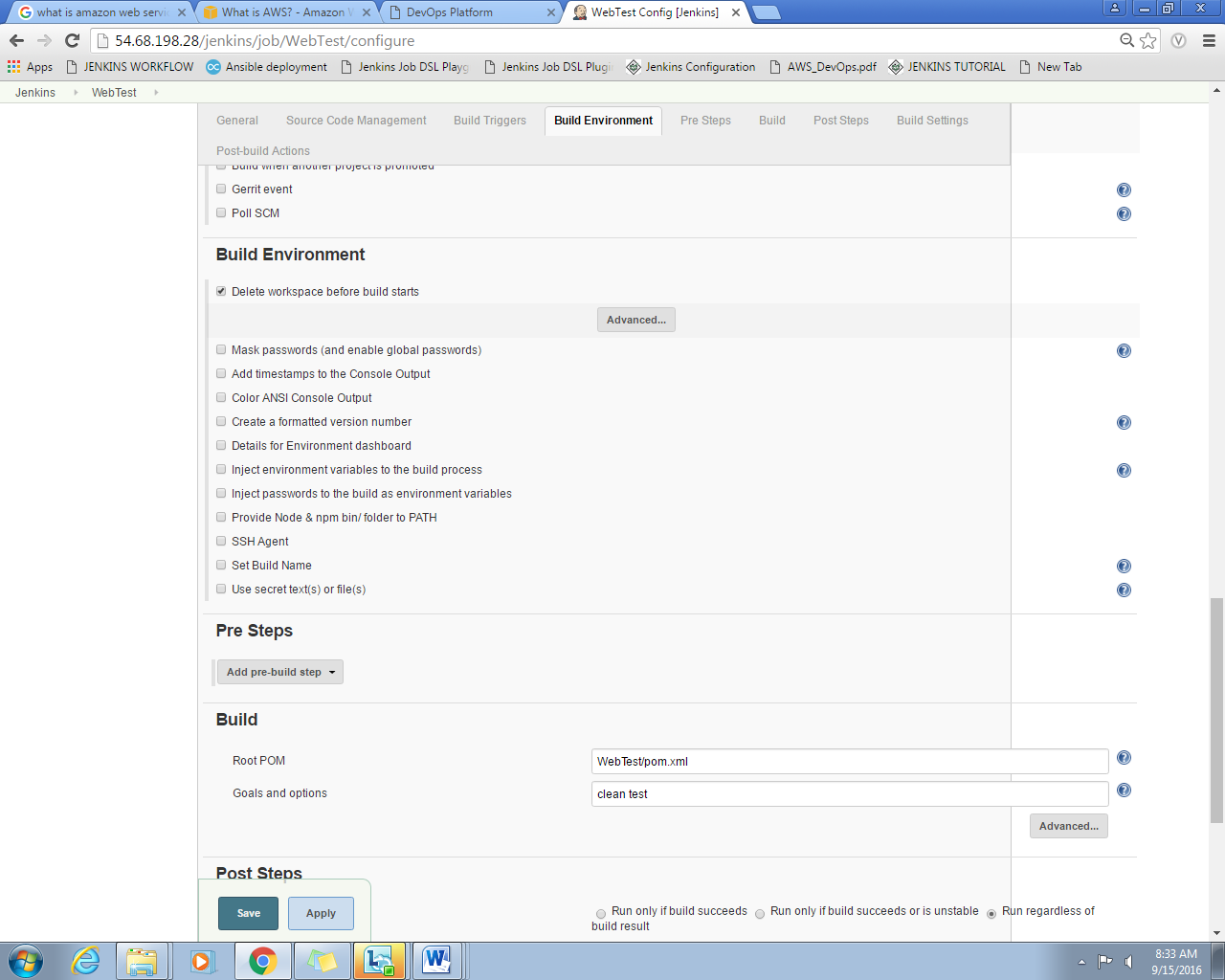
**Figure 2.28.** *General Configuration*



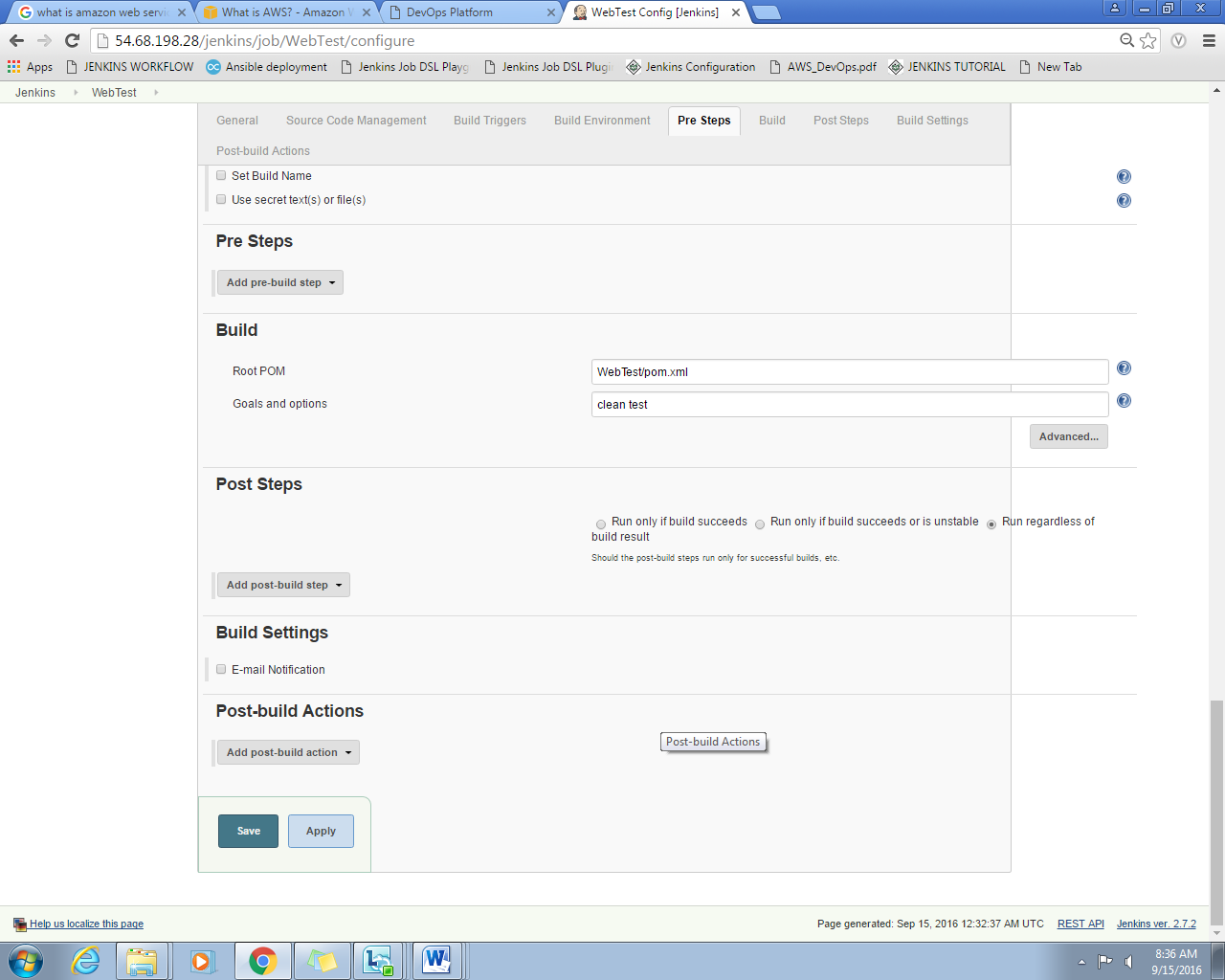
**Figure 2.29.** *Source Code Management Configuration*



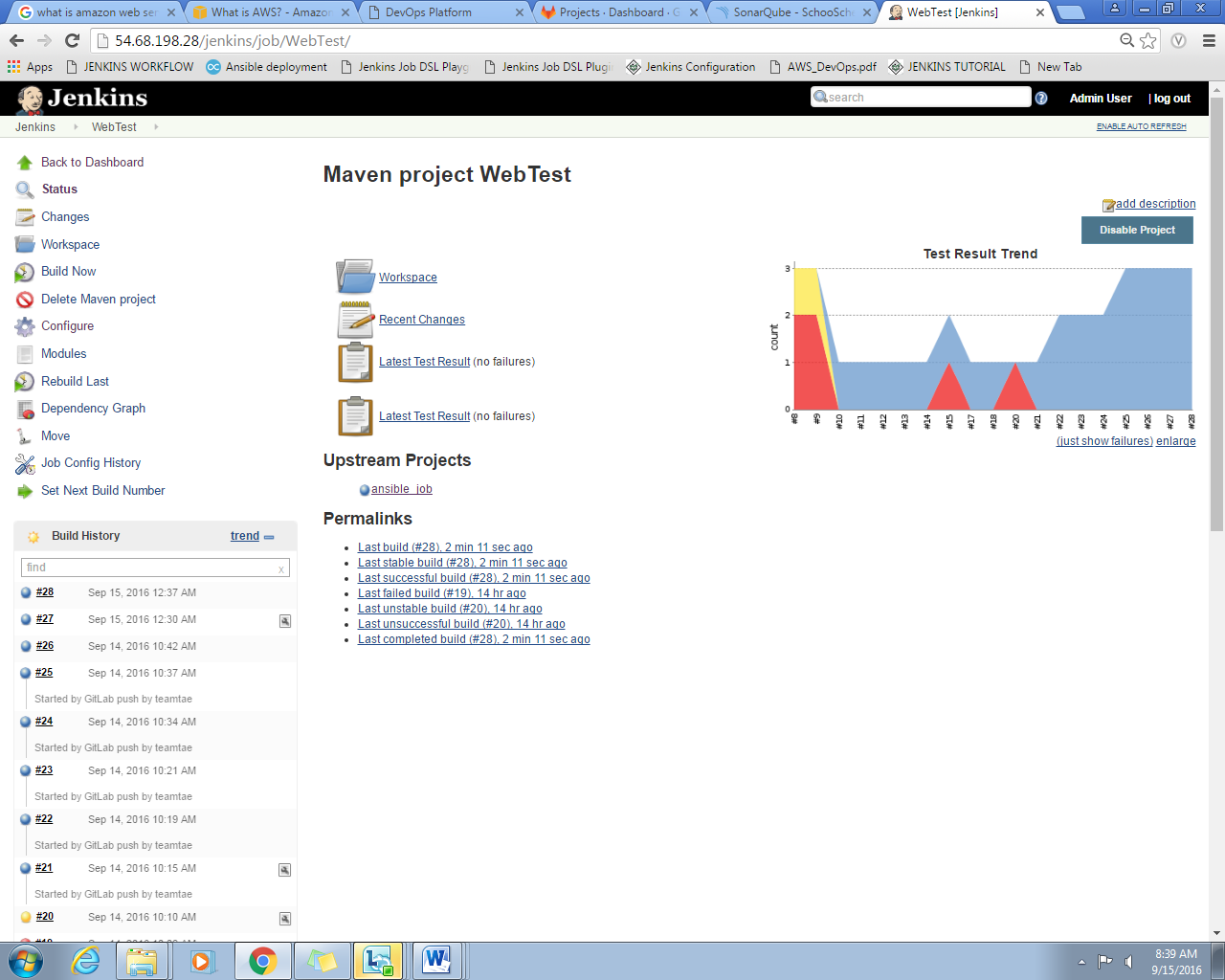
**Figure 2.30.** *Build Trigger Configuration*



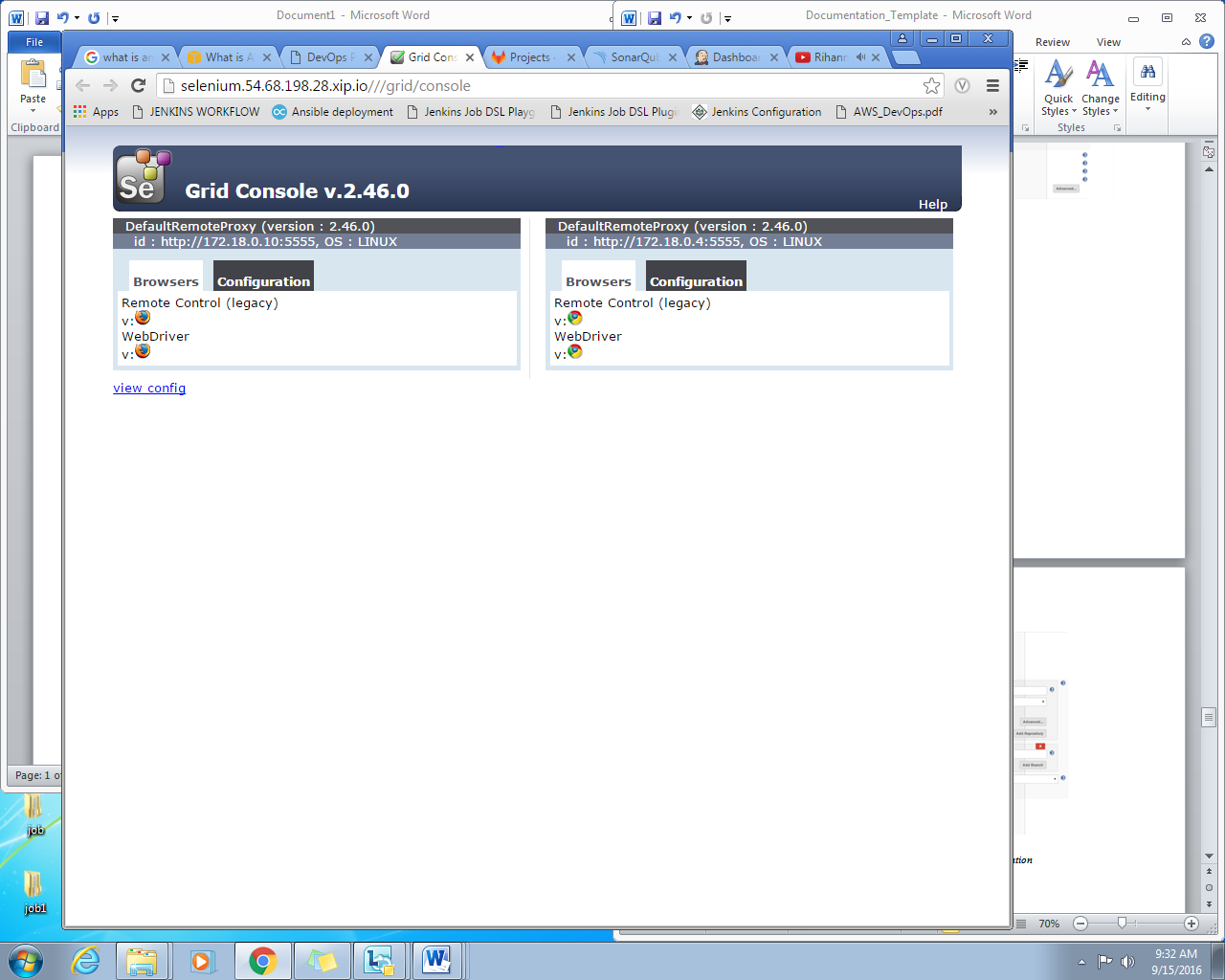
**Figure 2.31.** *Build Environment Configuration*



**Figure 2.32.** *Build Configuration*



**Figure 2.33.** *Selenium*



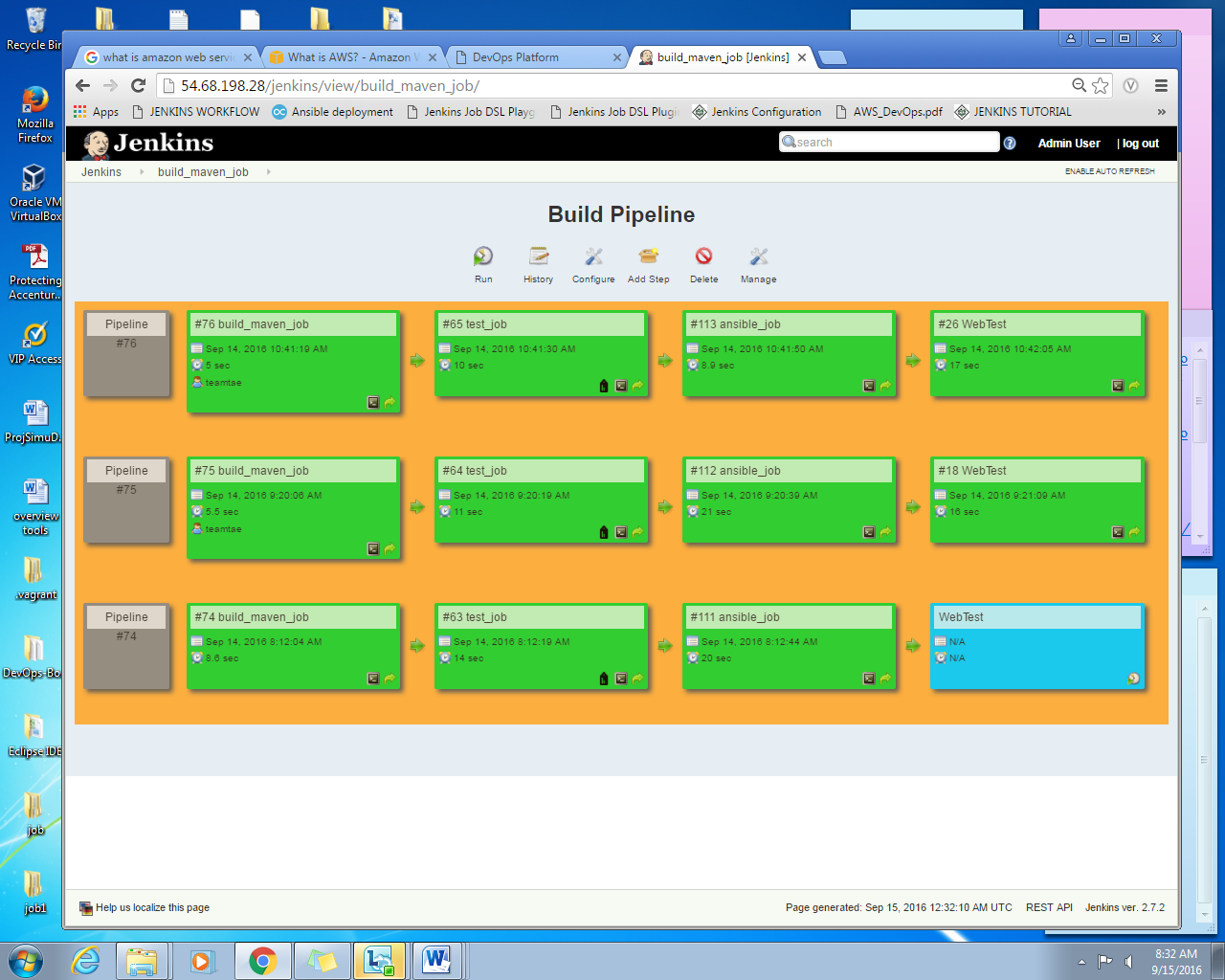
**2.4.8 GitLab**

The following projects that were pushed in the repository are listed in GitLab homepage.

 **Figure 2.34.** *Projects in GitLab*

**2.4.9 Pipeline**

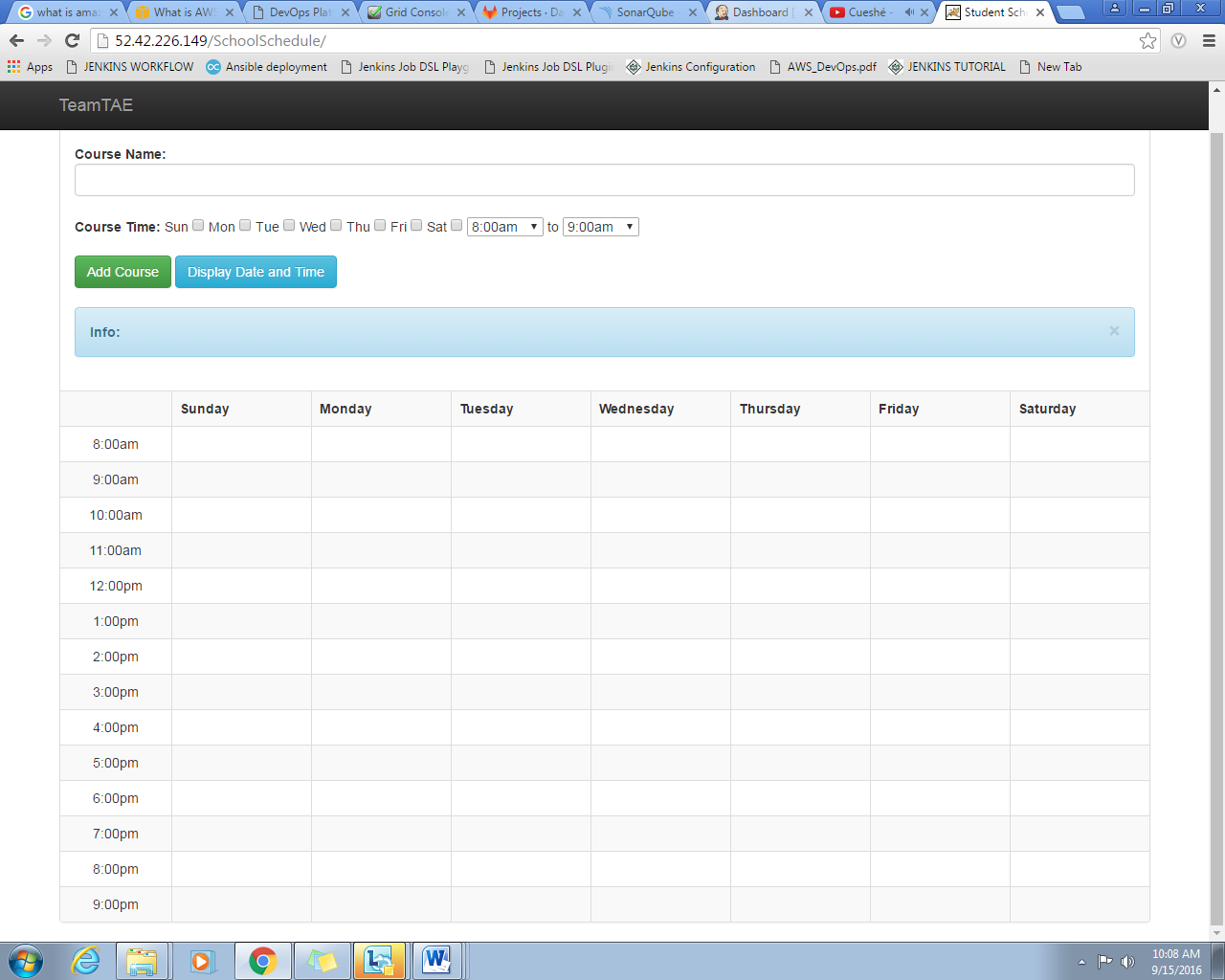
It is a diagram that shows the following stages of job. It shows whether building of jobs is successful or not. Green indicates successful job, yellow – job is building, blue- job is on cue, and red if job is not successful.



**Figure 2.35.** *Build Pipeline*

**2.4.10 Java based Web Application in Tomcat**

The web application can be access in tomcat webserver, which is <http://52.42.226.149/SchoolSchedule/>. This web application enables students to create a class schedule depending on what course, date and time they prefer.



**Figure 2.36.** *Web Application*

**CHAPTER 4 – SUMMARY AND CONCLUSION**

**Blahh… blahh.. blah…**

**WORK PLAN**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tasks** | **In-Charge** | 9/6 | 9/7 | 9/8 | 9/9 | 9/13 | 9/14 | 9/15 |
| **CI/CD** |  |  |  |  |  |  |  |  |
| Set kick-off meeting | All |  |  |  |  |  |  |  |
| Agree on objectives | All |  |  |  |  |  |  |  |
| **1.Provision Environment (Set-up)** |  |  |  |  |  |  |  |  |
| 1. Create a Stack | Cris Narca |  |  |  |  |  |  |  |
| 1. Launch an instance | Cris Narca |  |  |  |  |  |  |  |
| 1. Install Docker | Mari Cris Ayes / Anjhilian Cabrera |  |  |  |  |  |  |  |
| 1. Create and Deploy Tomcat | Mari Cris Ayes / Anjhilian Cabrera |  |  |  |  |  |  |  |
| **2.Java Web App** |  |  |  |  |  |  |  |  |
| 1. Provide Project | Mari Cris Ayes/Joey Robles |  |  |  |  |  |  |  |
| **3.Configuration** |  |  |  |  |  |  |  |  |
| 1. GitLab to Jenkins | Mari Cris Ayes / Anjhilian Cabrera |  |  |  |  |  |  |  |
| 1. Maven to Jenkins | Mari Cris Ayes / Anjhilian Cabrera |  |  |  |  |  |  |  |
| 1. Sonarqube to Jenkins | Mari Cris Ayes / Anjhilian Cabrera |  |  |  |  |  |  |  |
| 1. Ansible to Jenkins | Mari Cris/ Anjhilian/ Cris |  |  |  |  |  |  |  |
| 1. Tomcat to Jenkins | Mari Cris Ayes / Anjhilian Cabrera |  |  |  |  |  |  |  |
| 1. Selenium to Jenkins | Mari Cris Ayes / Anjhilian Cabrera |  |  |  |  |  |  |  |
| **4.Creation of Cartridges** |  |  |  |  |  |  |  |  |
| 1. Docker compose | Joey Robles |  |  |  |  |  |  |  |
| 1. Platform management | Pauleen Pineda |  |  |  |  |  |  |  |
| 1. Docker cartridge | Cris Narca |  |  |  |  |  |  |  |
| **5. Documentation** |  |  |  |  |  |  |  |  |
| 1. Document step by step process of provisioning with screens shots | Pauleen Pineda |  |  |  |  |  |  |  |
| 1. Document roles and responsibilities of each team member | Pauleen Pineda |  |  |  |  |  |  |  |
| 1. Document your work plan | Pauleen Pineda |  |  |  |  |  |  |  |
| 1. Document how SCM and CI works | Pauleen Pineda |  |  |  |  |  |  |  |
| 1. Overview of tools | Pauleen Pineda/Joey Robles |  |  |  |  |  |  |  |

References:

* <http://view.dckr.info/DockerIntro.pdf>
* <https://en.wikipedia.org/wiki/PuTTY>
* <https://theagileadmin.com/what-is-devops/>
* <https://d0.awsstatic.com/whitepapers/AWS_DevOps.pdf>
* <https://about.gitlab.com/about/>
* <http://www.tutorialspoint.com/jenkins/>
* <https://maven.apache.org/what-is-maven.html>
* <http://www.sonarqube.org/>
* <https://www.ansible.com/how-ansible-works>
* <http://tomcat.apache.org/>
* http://www.seleniumhq.org/docs/01\_introducing\_selenium.jsp