**Project (Industry Project 1)**

1. **Achieved Goal**

Implement CI/CD for ABC Company is able to be—

▪ highly available

▪ highly scalable

▪ highly performant

▪ easily built and maintained

▪ developed and deployed quickly

1. **Data Flow Architecture/Process Flow**

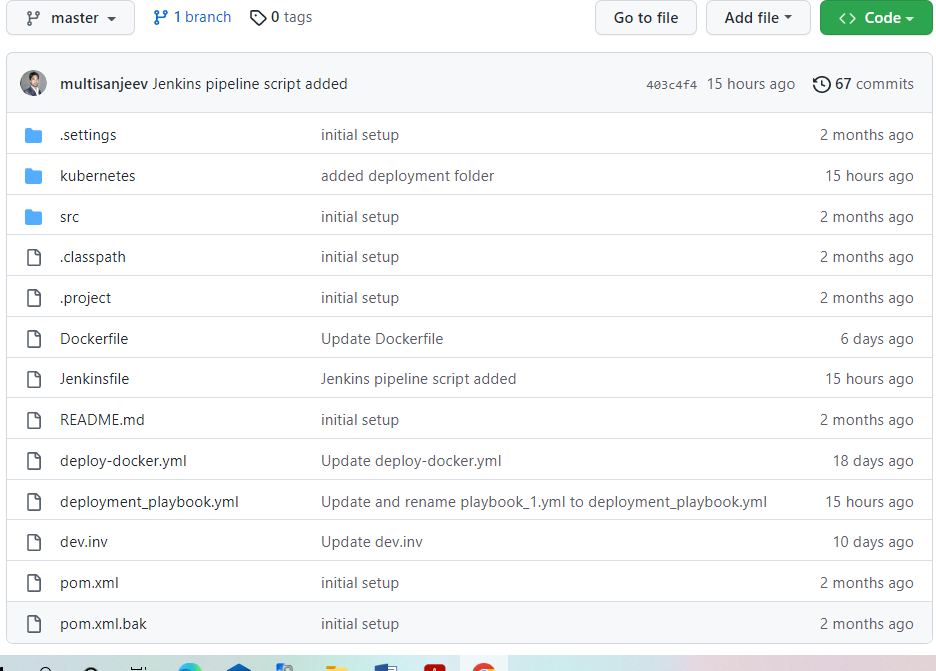


1. **GIT – [ Used for code versioning]**

GIT repository URL – [**https://github.com/multisanjeev/industry\_grade\_1.git**]

* Created GIT Repository for code management
* Add and commit the codebase on local GIT Repo.
* Add origin [ git remote add origin https://github.com/multisanjeev/industry\_grade\_1.git]
* Pushed codebase on git [ git push origin master]



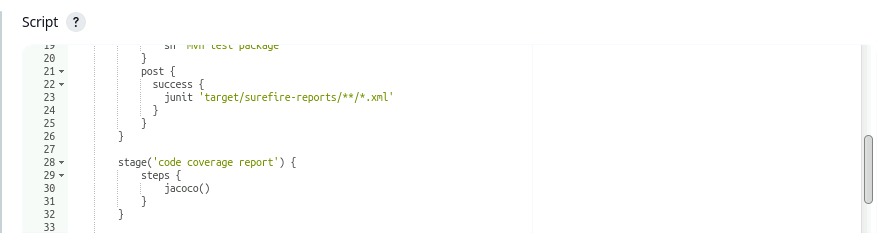


1. **Maven [Used for project compile, Test and package]**

* Compile – It compiles the source code, converts the java files to class and stores the classes in target/classes folder. [ mvn compile]
* Test – It runs unit tests for the project. [ mvn test]
* Package - This step packages the compiled code in distributable format like JAR or WAR. [mvn package]

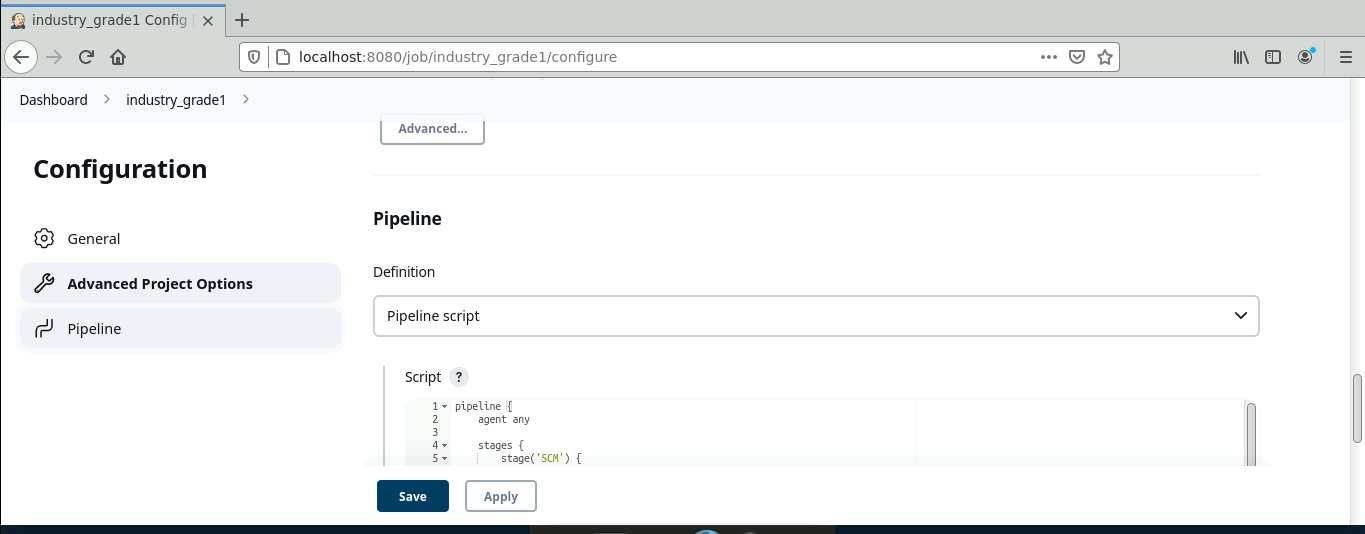


Jacoco – [For code coverage report] =>



1. **Jenkins [ used for CI/CD pipeline]**

* Added agent “Slave” for run the pipeline on slave machine.
* Installed Ansible plugin for Ansible configuration with Jenkins.
* Added Ansible path under “global configuration”
* Created job with pipeline.



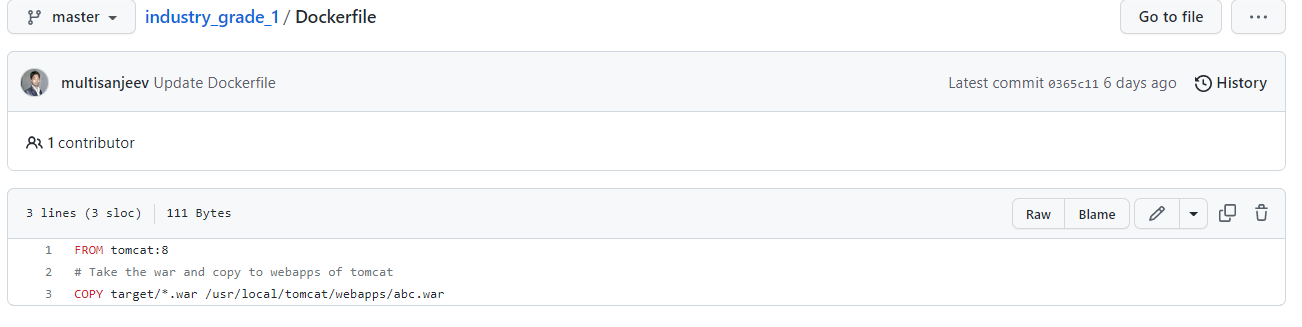
1. **Docker [Used for build and push image on Docker hub]**

* Added Docker hub login credentials under “manage credential section”
* Used variable for manage Docker image version dynamically.
* Generate Docker image [ Docker build -t docker\_username/image:tag ]
* Used Jenkins “with credential” function for Docker login

[ Docker login -u user\_name -p password\_variable]

* Pushed image on Docker hub [ docker push image\_name]



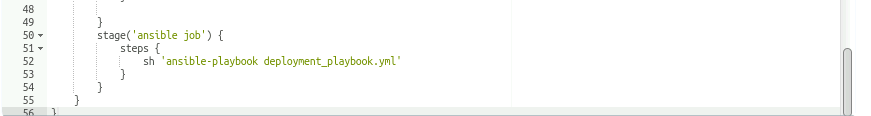


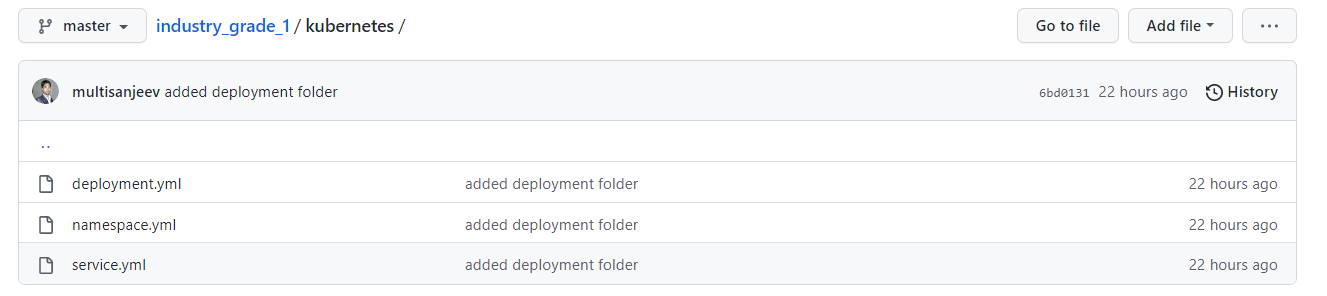
1. **Ansible – [ Used for configuration management and Deployment]**

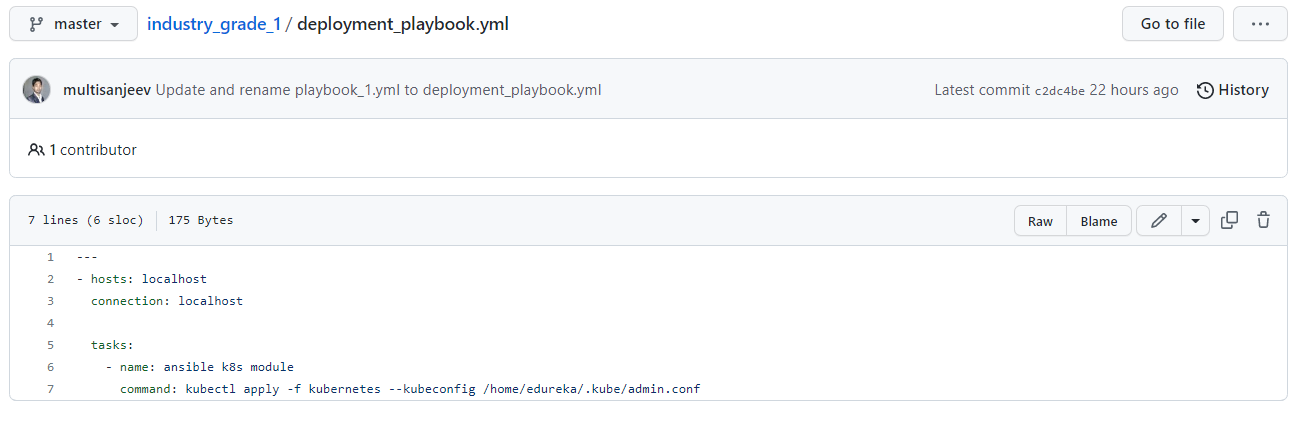
* Used Ansible playbook for deployment
* Run Ansible playbook on localhost, because of we have 2 machine master and slave. If we run the job on slave machine, then we can’t deploy project through kubernetes. Kubernetes needs nodes for pod, namespace, deployment and service.

In our case kubernetes master step there on master machine, so we ran the Ansible playbook on localhost.

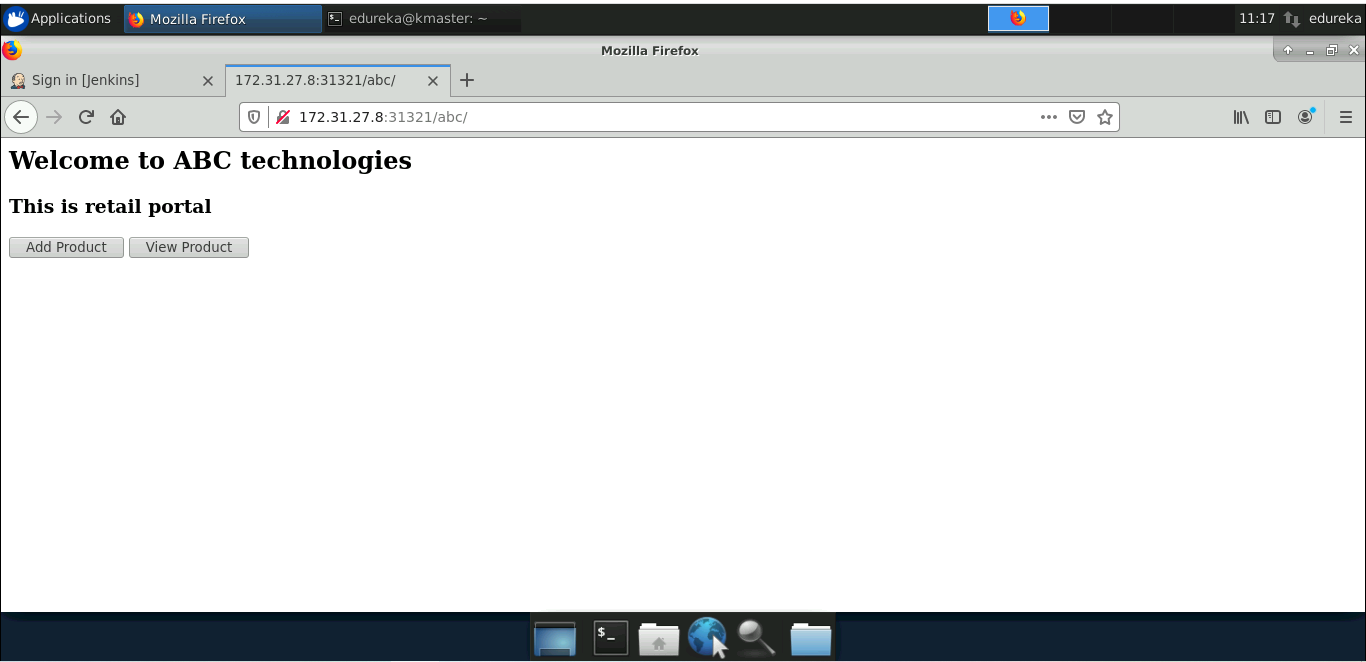
* Kubernetes deployment used Pod, namespace, replica set, deployment and services.
* Used NodePort service type of expose the application outside word.





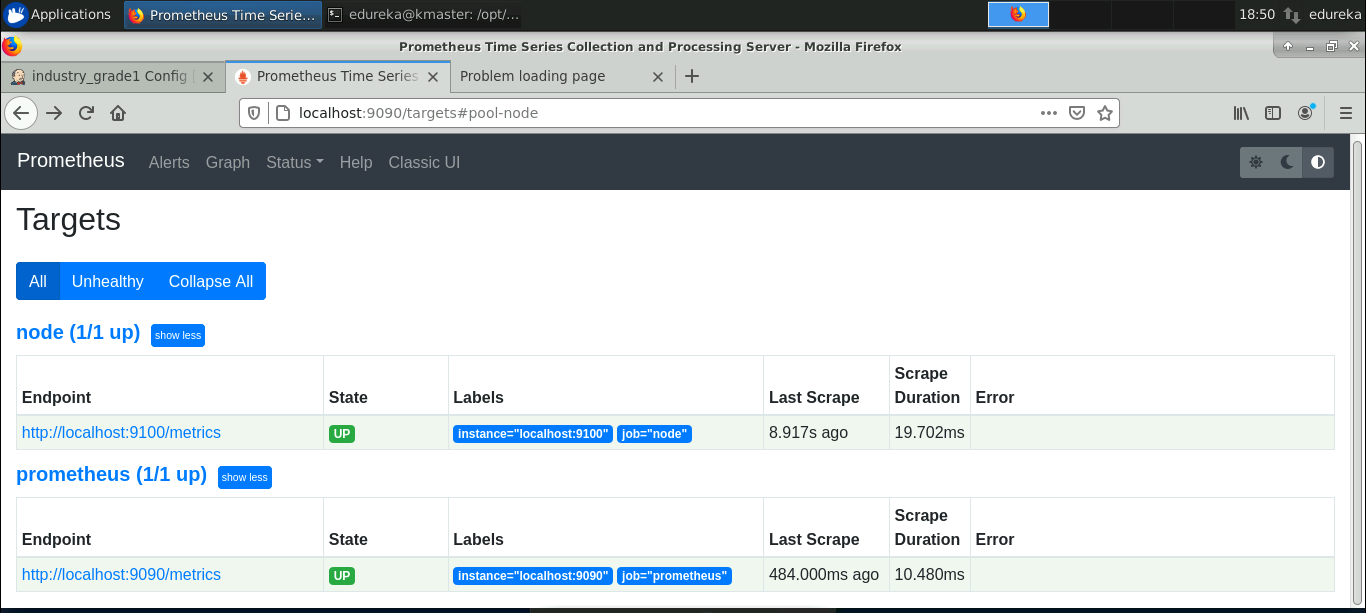


* Successfully application deployed with the help of kubernetes and Ansible playbook.
* **Running application screenshot**.

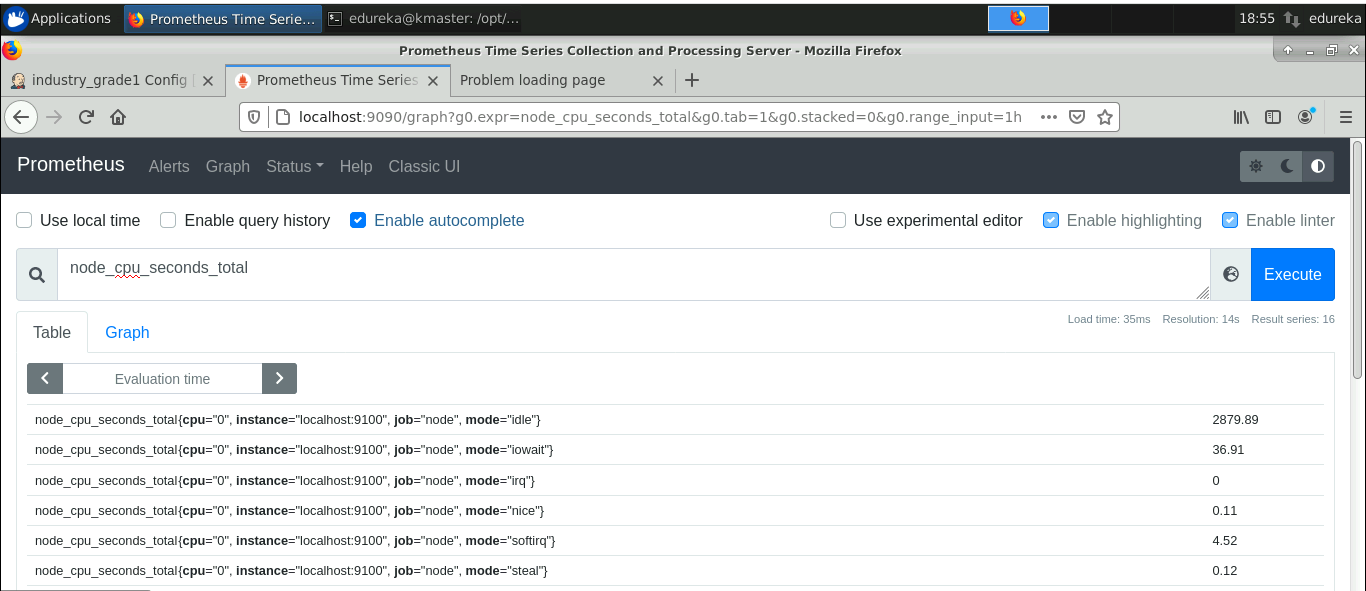


**Monitoring Tool:**

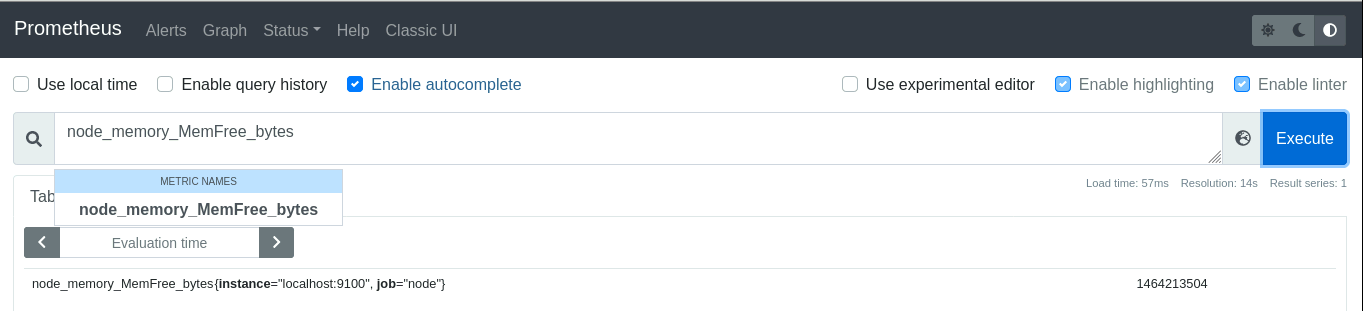
1. **Prometheus - [ Running on port 9090]**



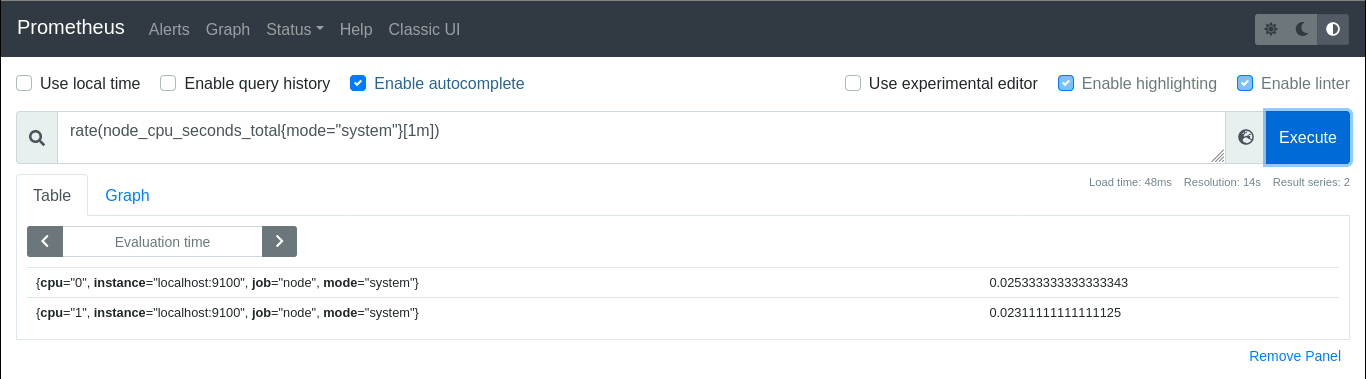
**CPU utilization Prometheus:**



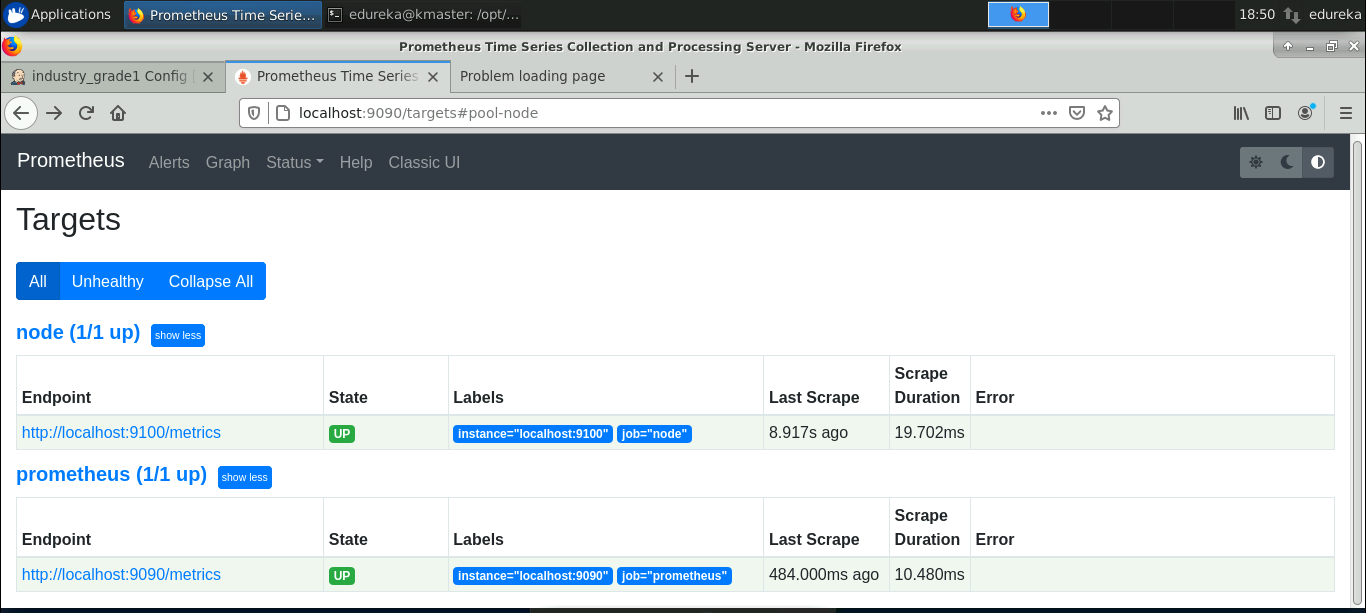
**Node Memory free in bytes:**



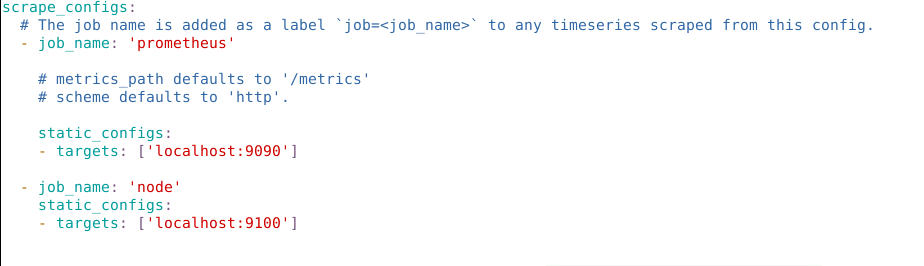
**Node CPU per second:**



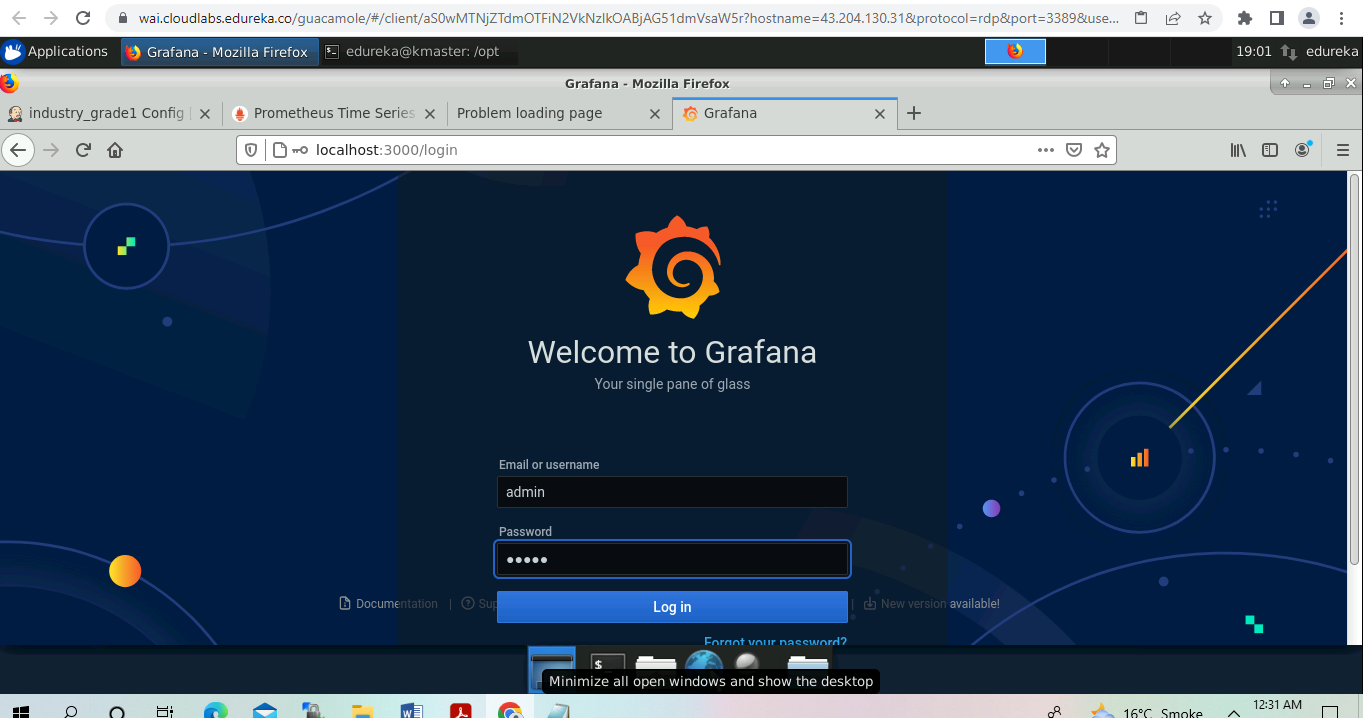
1. **Node Exporter – [Running on port 9100]**



**Node exporter configuration added on Prometheus.yml file:**

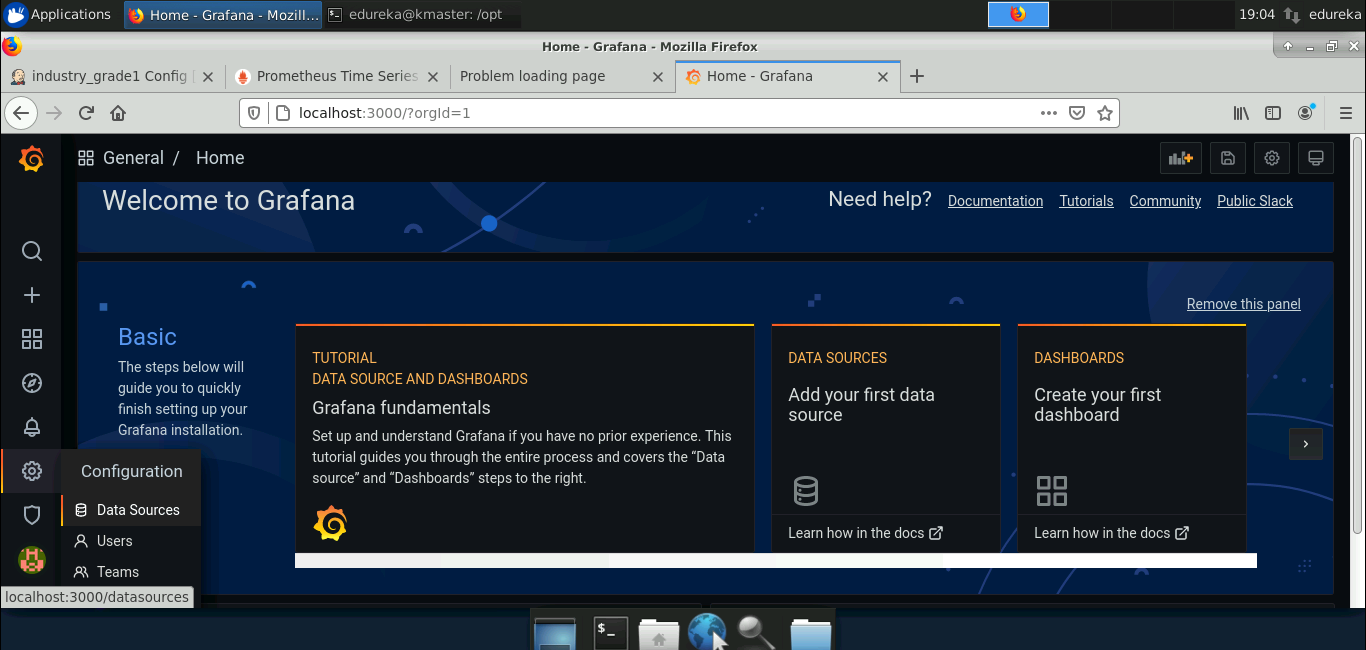


**Grafana: [Default port : 3000]**

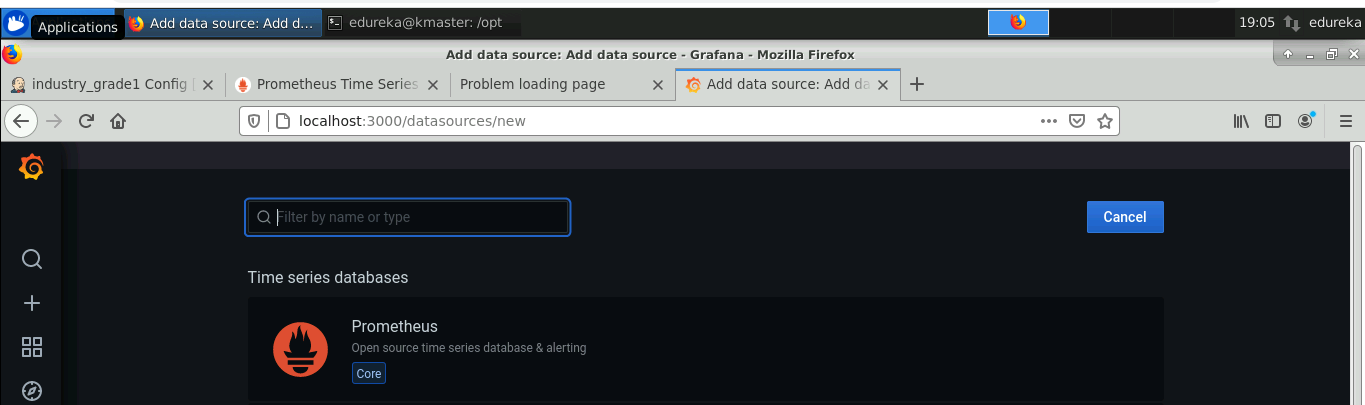


**Grafana + Prometheus configuration:**

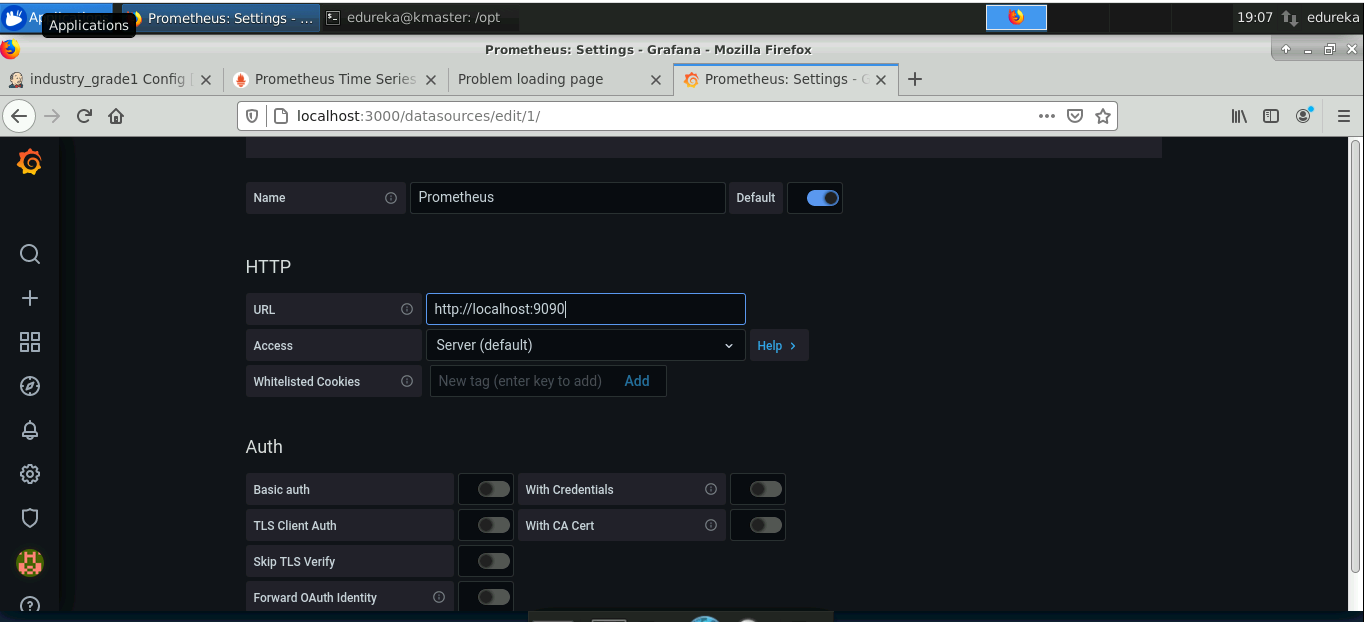
1. **Add data source**



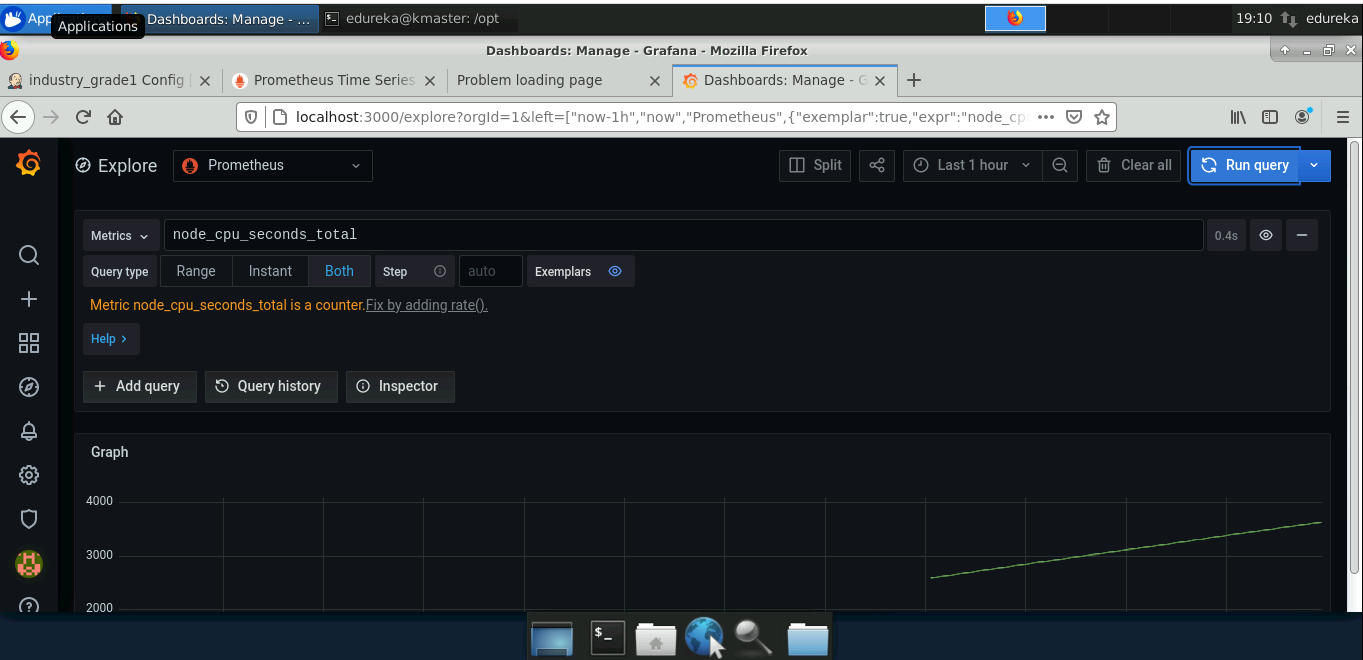
1. **Select Prometheus.**



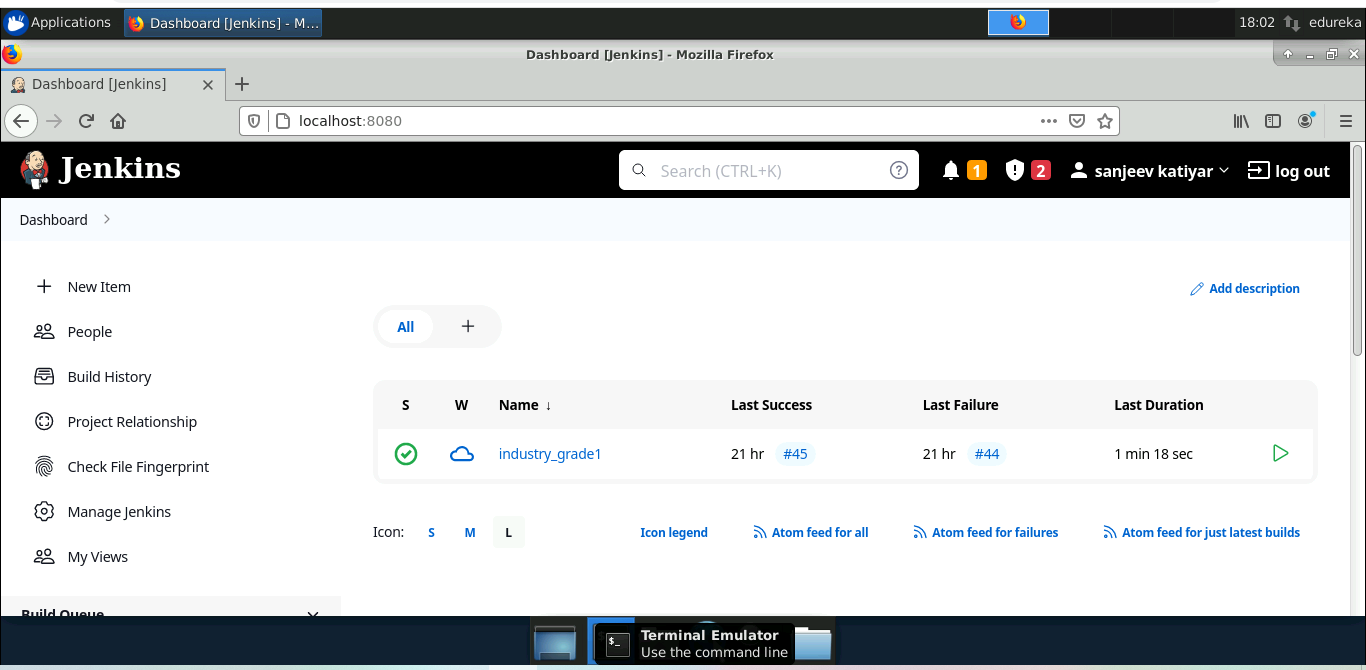
1. **Add Prometheus URL and save**



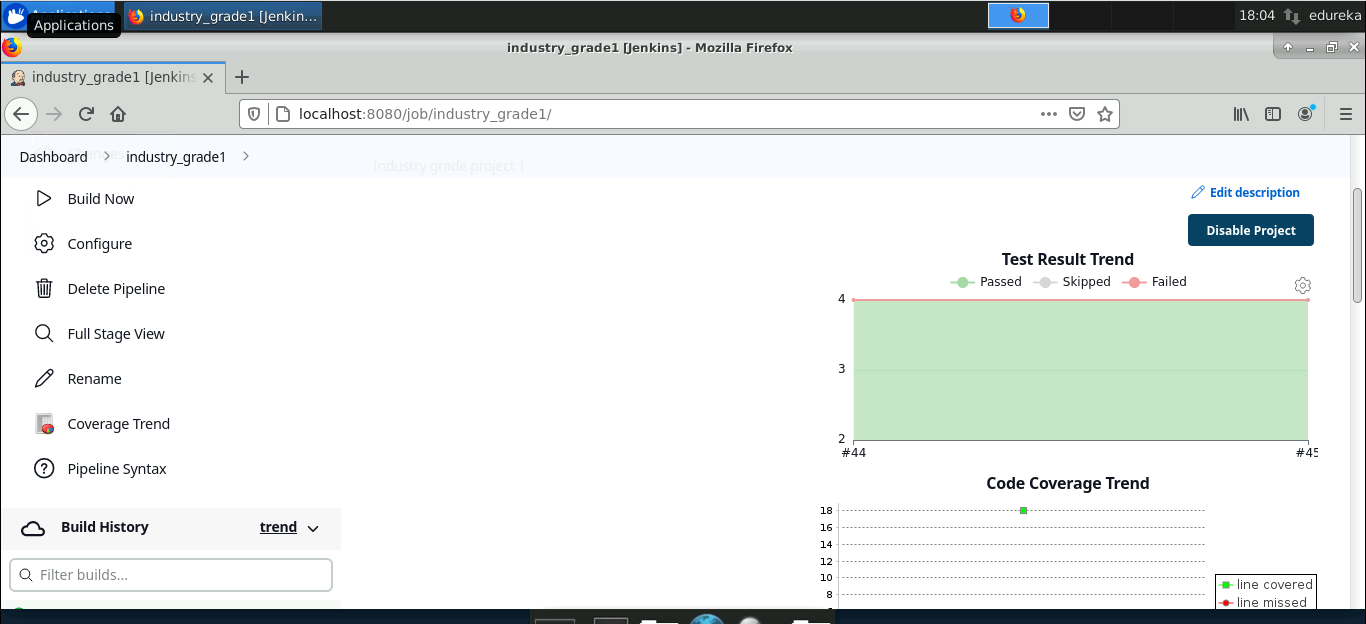
1. **CPU utilization Grafana screenshot:**



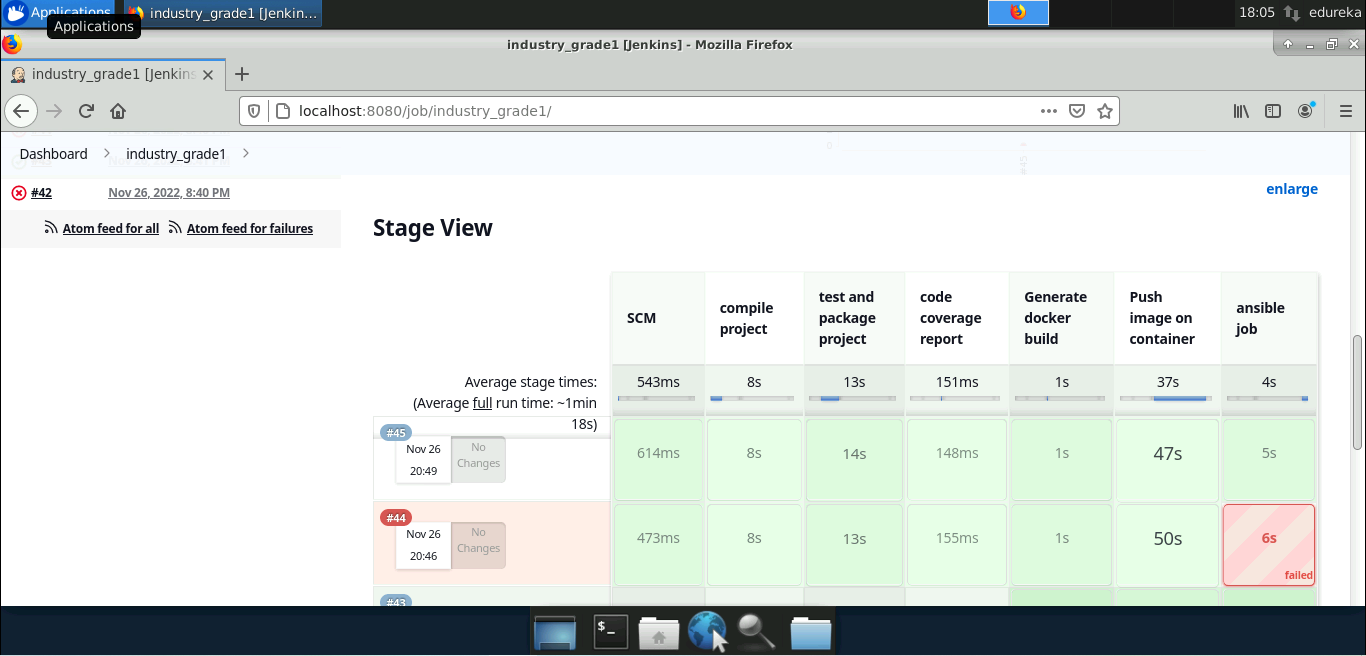
**Jenkins Screenshot:**



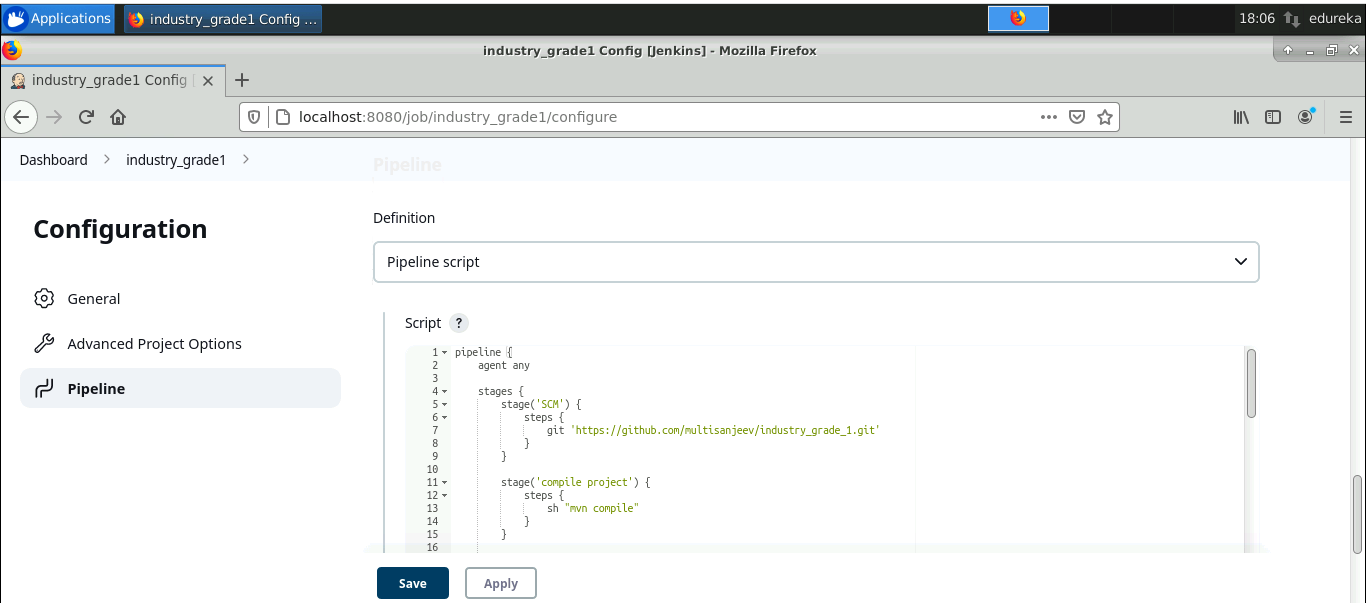
**Unit Test and Jacoco (code coverage) Screen:**



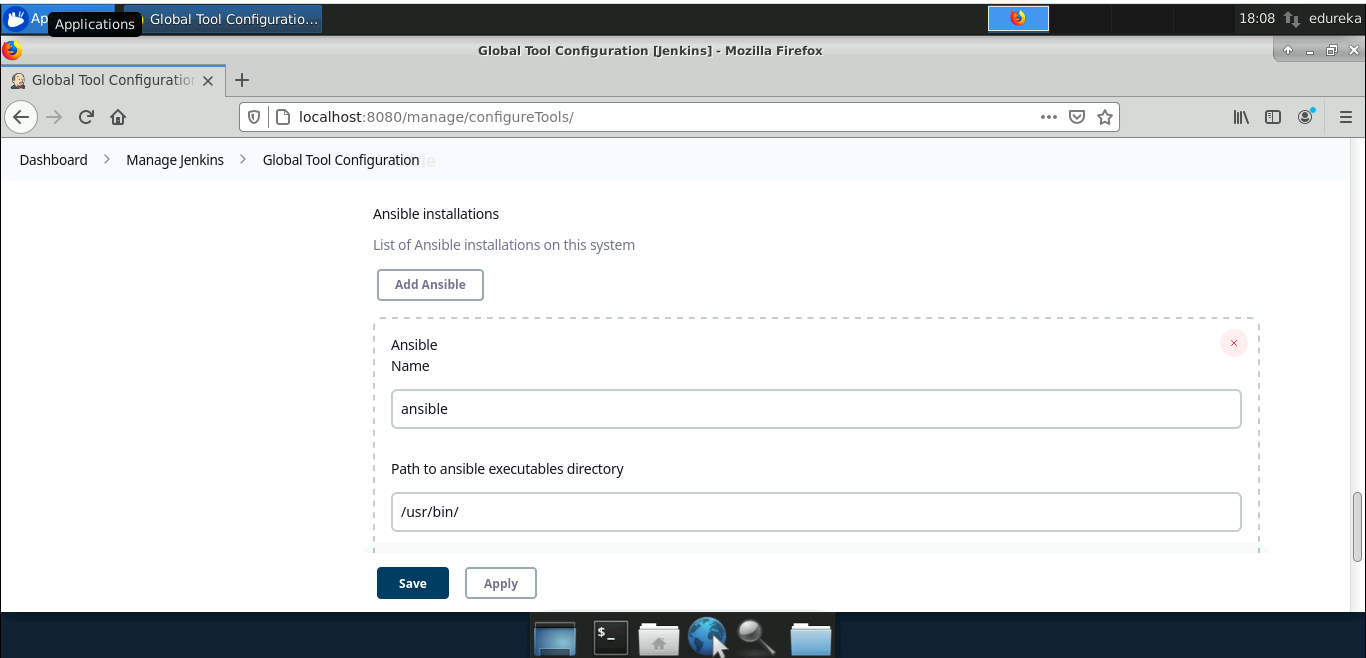
**Pipeline execution steps:**



**Pipeline configuration:**



**Jenkins + Ansible configuration:**



**Manage Credentials:**

