***NAME : AKASH KAMLAKAR HADGE***

***BATCH : Prog\_Jas21\_DevOps\_1***

***WEEK-8 ASSIGNMENT***

***----------------------------------------------------------------------------------------------------***

**Assignment-1 Marks: 20**

Explain the architecture of Ansible along with its use case and architectural diagram.

What is Ansible?

Ansible is a simple, open-source, IT automation engine that automates cloud provisioning, intra-service orchestration, configuration management, application deployment, and many other IT needs in day to day life in Software engineer.

It is designed for multi-tier deployments. Thereby it aligning with the cloud’s basic structure precisely. Ansible models your IT infrastructure by describing how all of your systems inter-relate, rather than just managing one system at a time.

Automation simplifies complex tasks, not just making developers’ jobs more manageable but allowing them to focus attention on other tasks that add value to an organization. In other words, it frees up time and increases efficiency.

Why do we like Ansible?

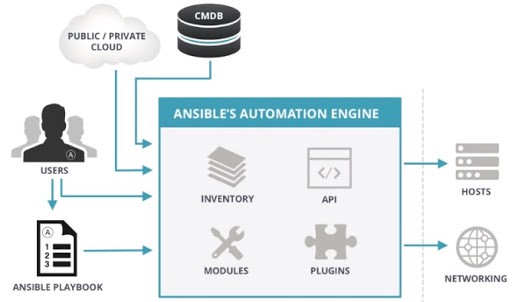
Ansible helps in modeling the cloud IT infrastructure with a description of inter-relation between all systems. Ansible does not implement any agents or additional custom security infrastructure. Furthermore, Ansible also offers flexibility for deployment and uses the simple YAML language in its Playbooks. Therefore, the user could easily describe automation jobs in simple English.

Advantages of Ansible

* **Free**: Ansible is an open-source tool.
* **Very simple to set up and use**: No special coding skills are necessary to use Ansible.
* **Powerful**: Ansible lets you model even highly complex IT workflows.
* **Flexible**: You can orchestrate the entire application environment no matter where it’s deployed. You can also customize it based on your needs.
* **Agentless**: You don’t need to install any other software or firewall ports on the client systems you want to automate. You also don’t have to set up a separate management structure.
* **Efficient**: Because you don’t need to install any extra software, there’s more room for application resources on your server.

Ansible Architecture

The Ansible orchestration engine interacts with a user who is writing the Ansible playbook to execute the Ansible orchestration and interact along with the services of private or public cloud and configuration management database. The below diagram explains it.



**Playbooks** here actually define your workflow because whatever tasks that you write in a playbook, it gets executed in the same order that you have written them. They are written in YAML format, which describes the tasks and executes through the Ansible. Also, you can launch the tasks synchronously and asynchronously with playbooks.

The above architecture has a bunch of **Host** machines to which ansible server connects and pushes the playbooks through SSH. It is not always needed to use an SSH for connecting with your host machines; you can also use a connection plug-in. For example, Ansible provides you with a [docker container connection plugin](https://docs.ansible.com/ansible/latest/scenario_guides/guide_docker.html) and using that connection plug-in, you can easily connect to all your Docker containers and start configuring right away.

It has an Ansible automation engine using which users can directly run a **playbook** that gets deployed on the hosts. There are multiple components in the Ansible automation engine. The first is a host inventory. It’s a list of all the IP addresses of all the hosts. Let’s go one by one, in the Ansible automation engine.

Ansible works against **multiple systems**in your infrastructure at the same time. It does this by selecting portions of systems listed in Ansible’s inventory file, which defaults to being saved in the location /etc/ansible/hosts. You can specify a different inventory file using the -i <path> option on the command line.

**Assignment-2 Marks: 30**

Create 2 new ansible nodes as per below specifications

IP Host name OS

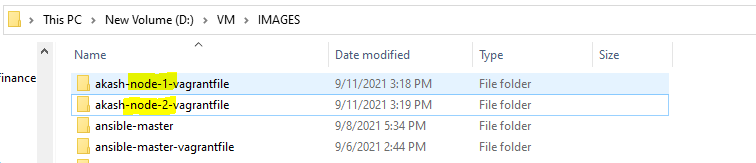
192.168.56.10 puneet-node-1 ubuntu 18.04

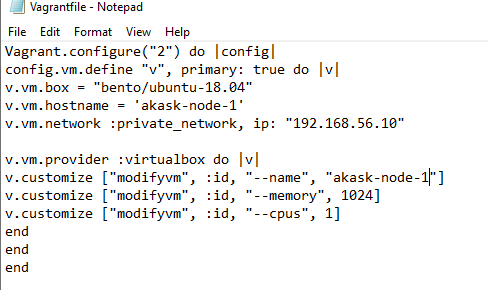
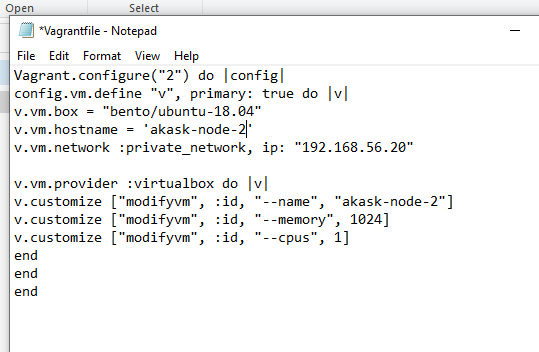
192.168.56.20 puneet-node-2 ubuntu 18.04

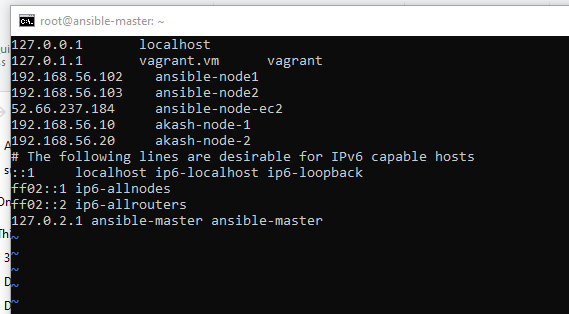
Create a single playbook to install

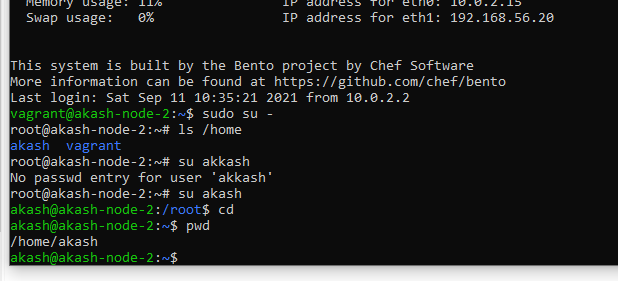
* apache2 on 192.168.56.10 with default webpage puneet.html with the contents “Welcome Puneet to the world of Apache with Ansible” with handlers and notify.
* nginx on 192.168.56.20

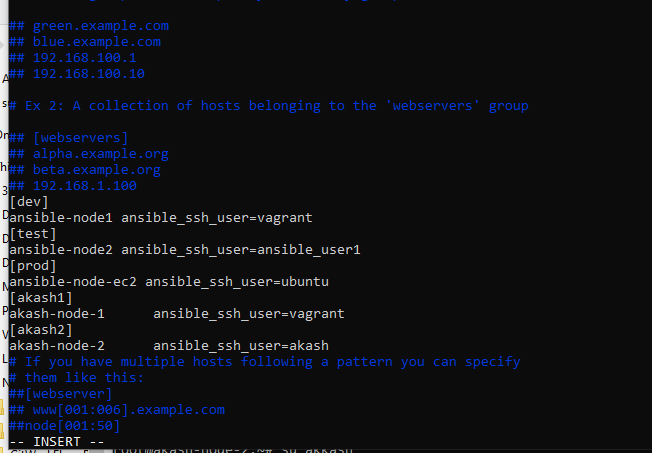
Note: Please replace the file names with your respective name in the given examples.

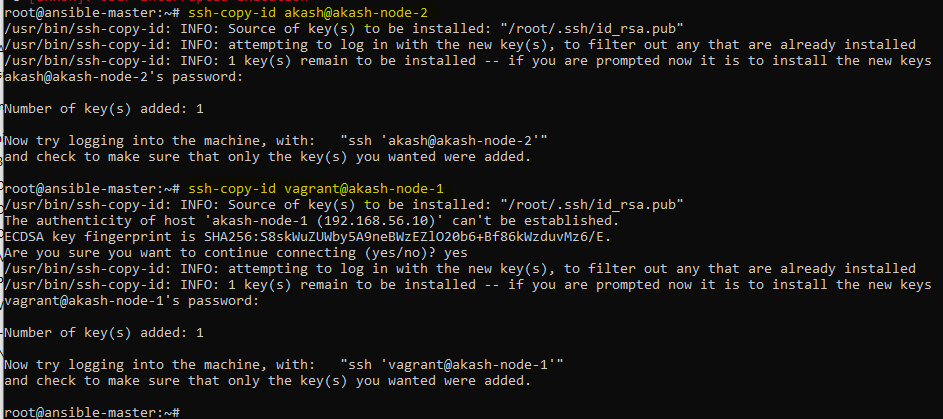


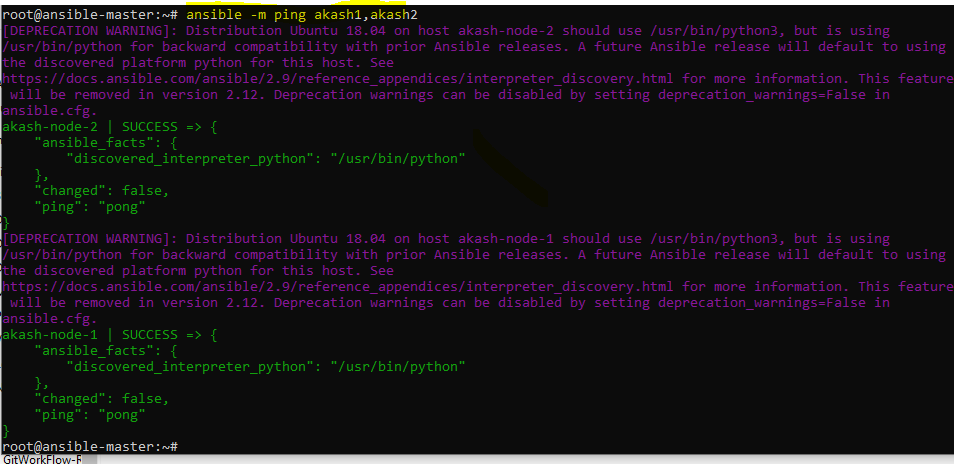
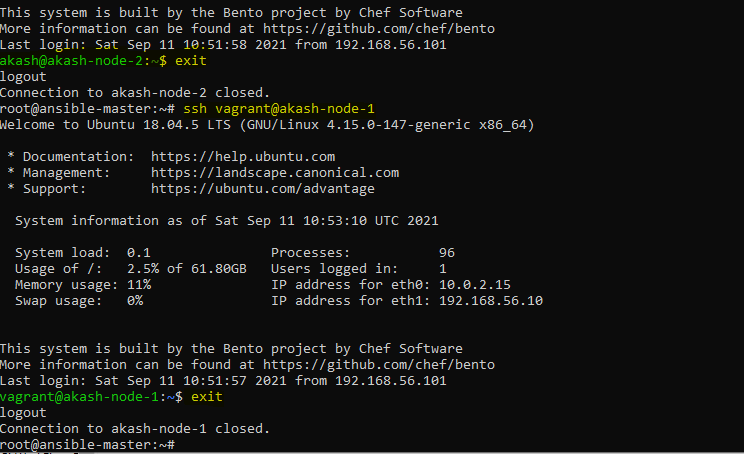
 

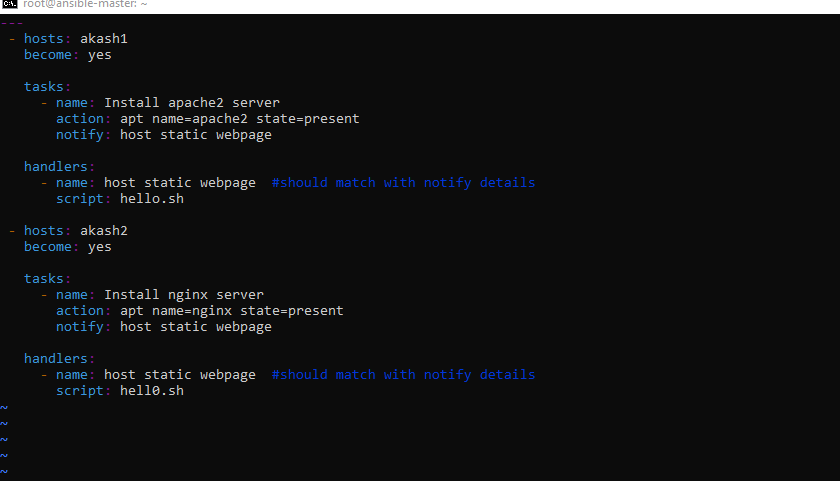


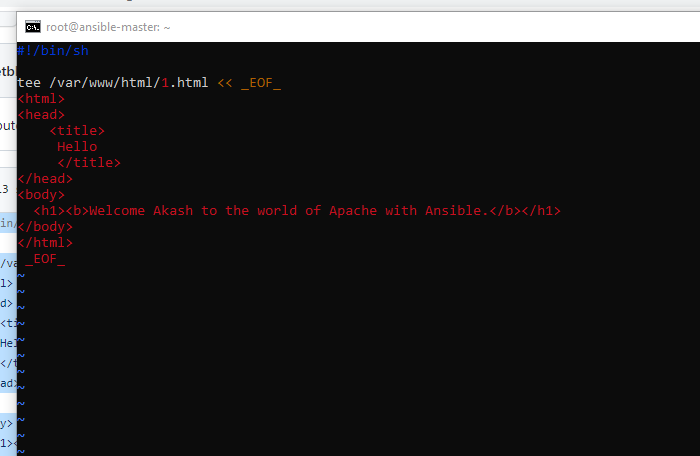


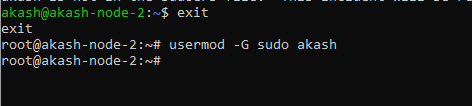


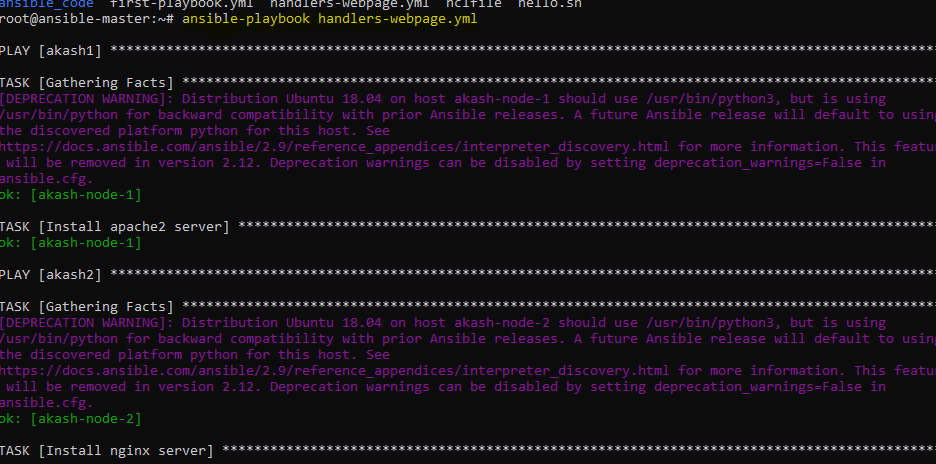


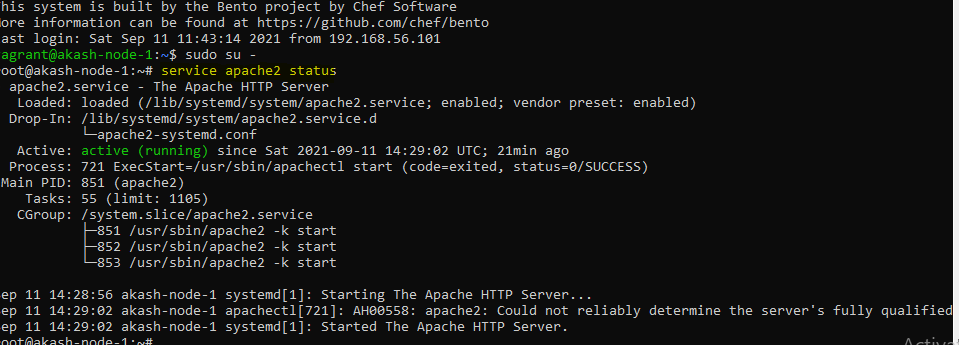
 

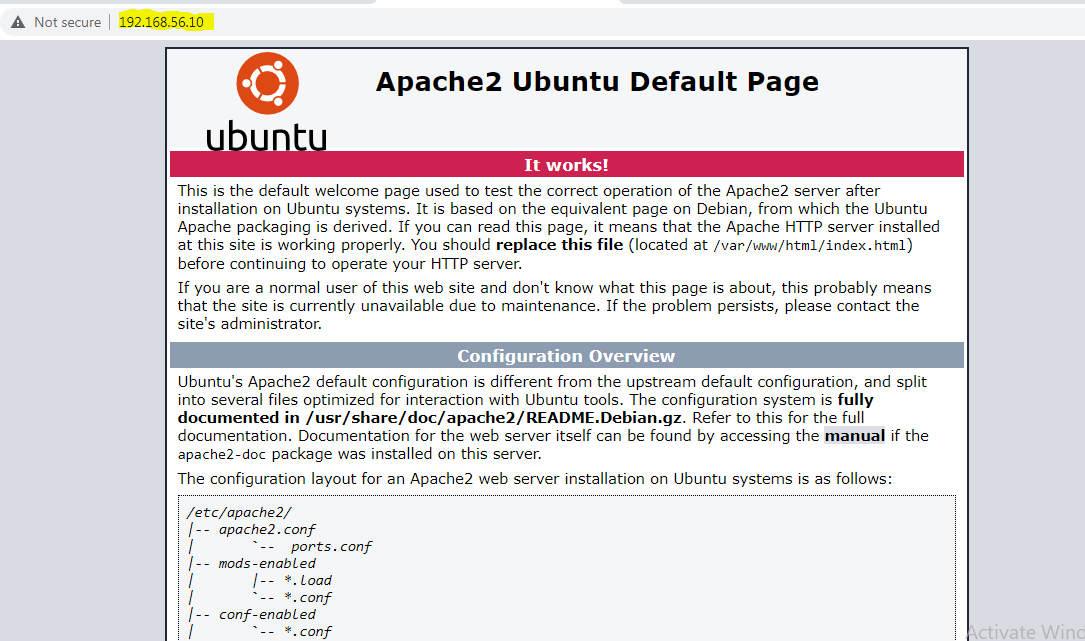


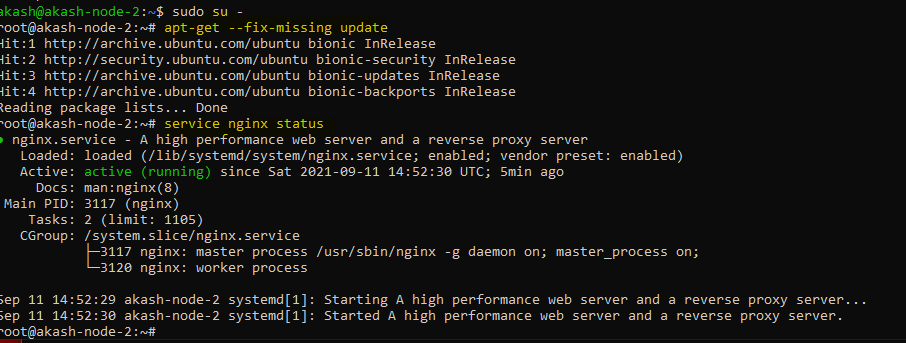


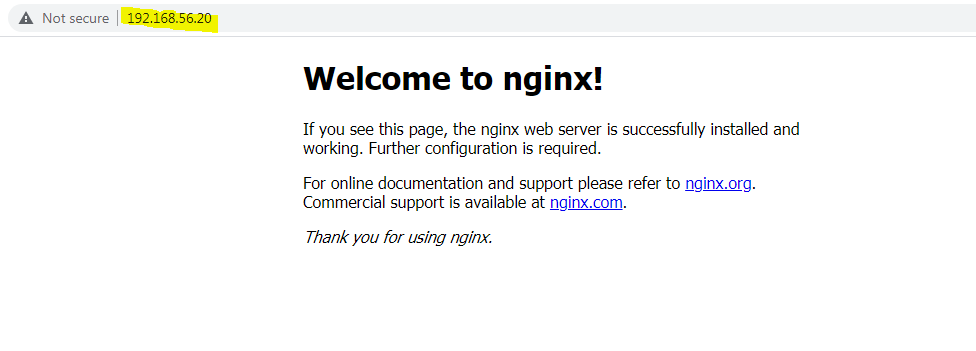


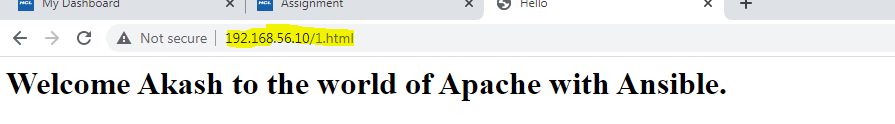


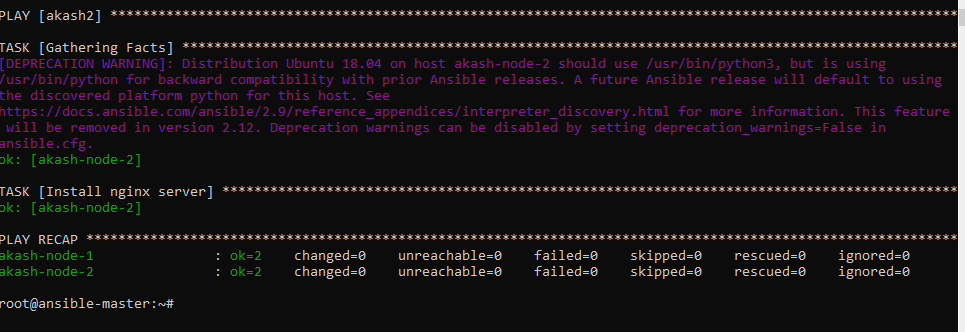






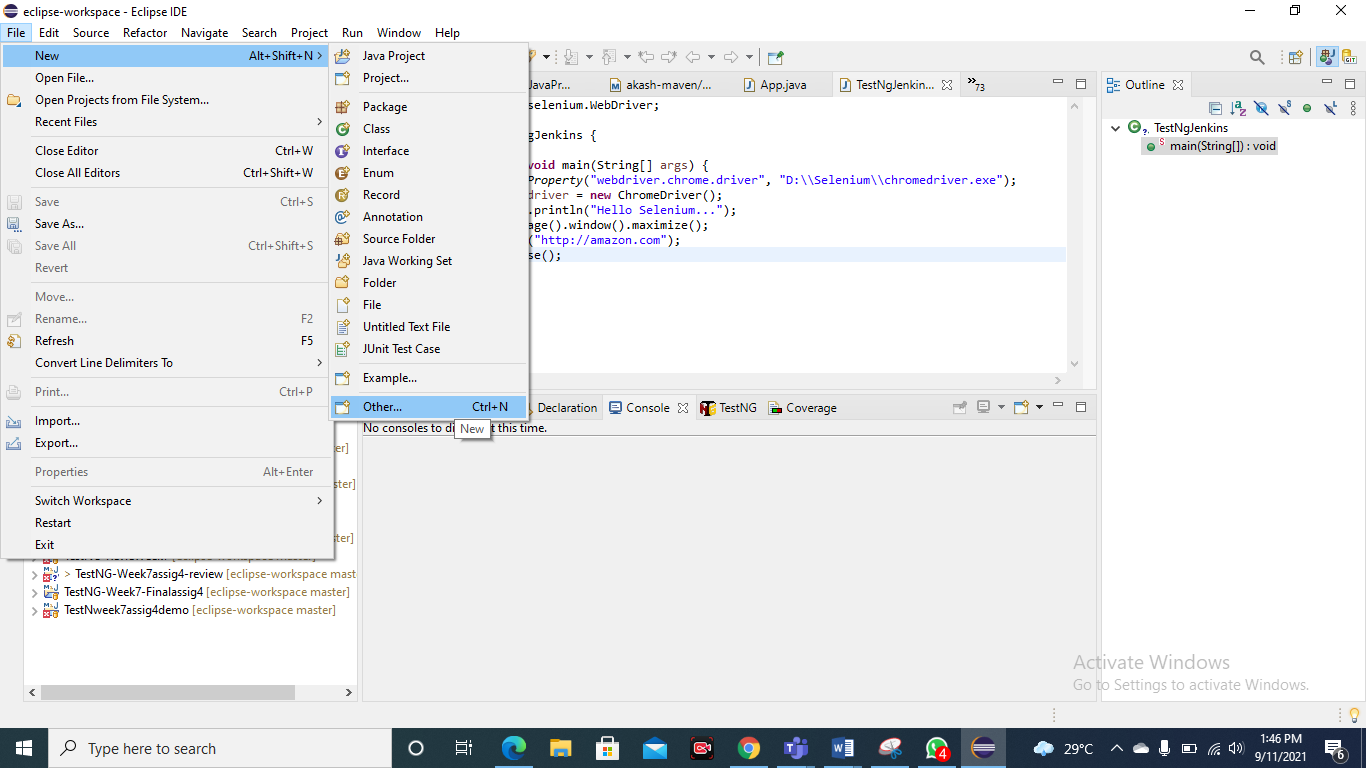


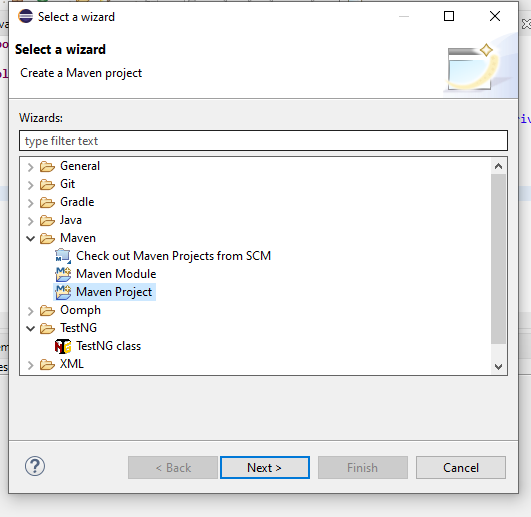


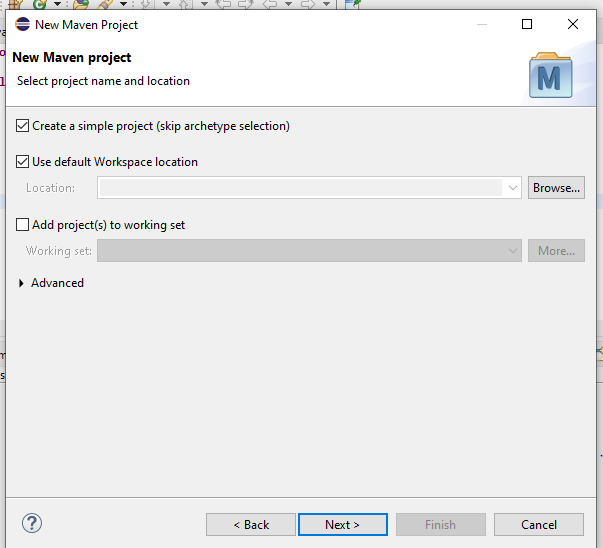


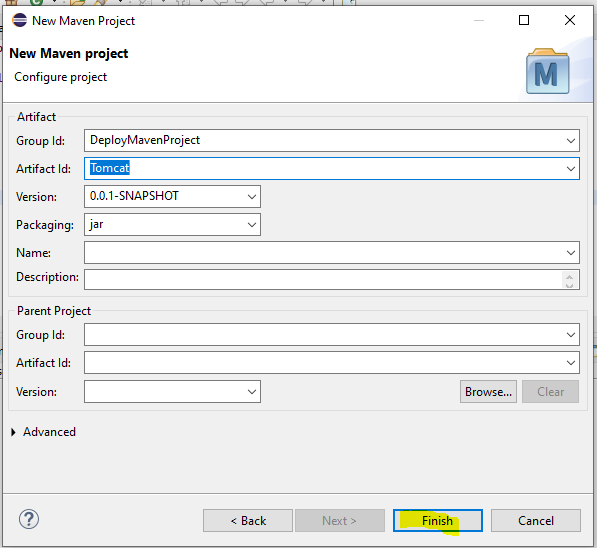
**Assignment-3 Marks: 25**

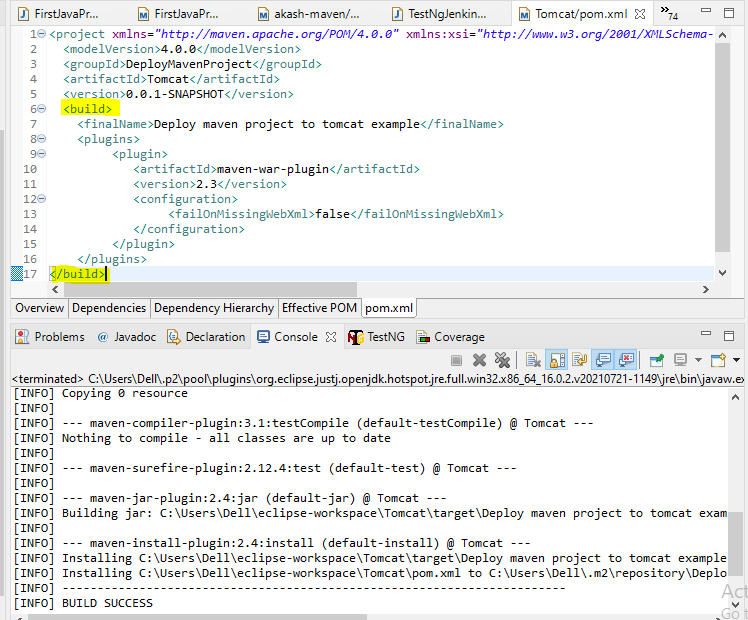
Create a maven-based Java project and deploy it to tomcat container.

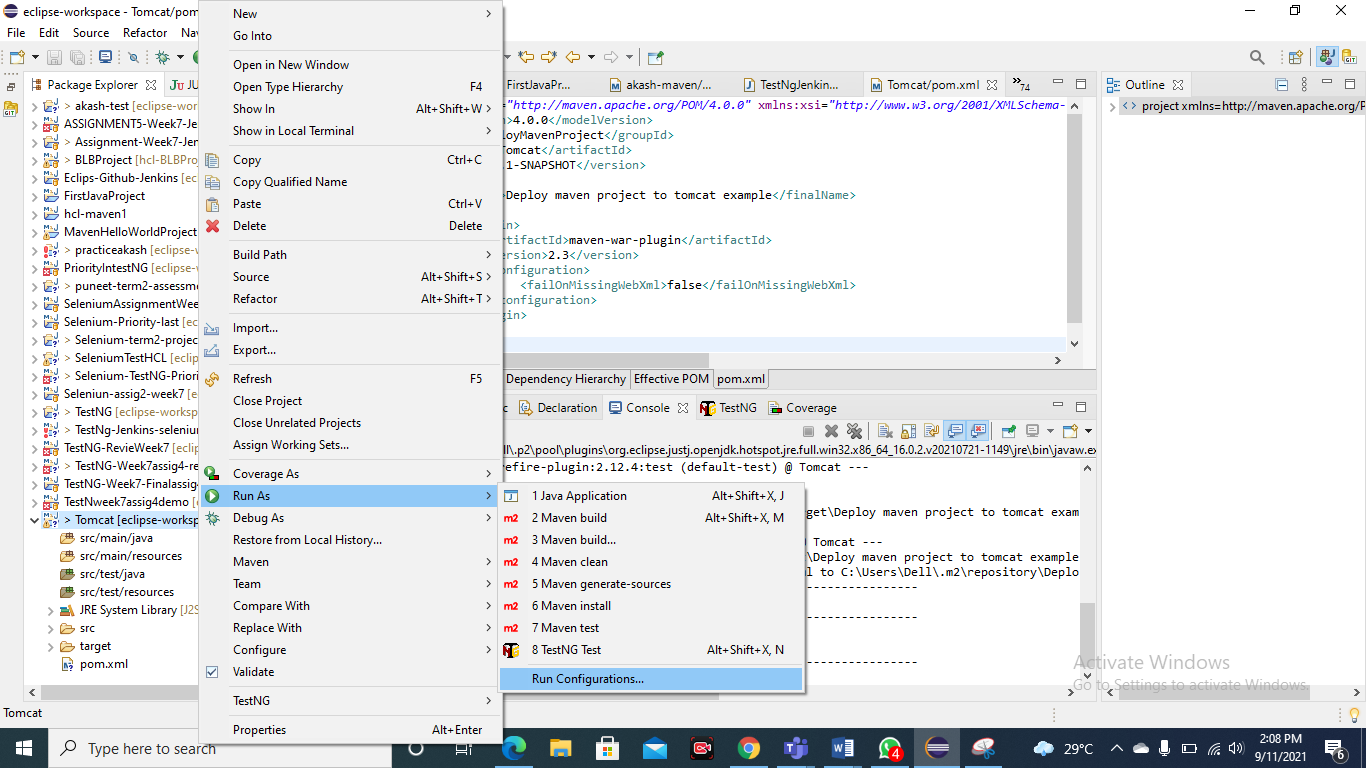


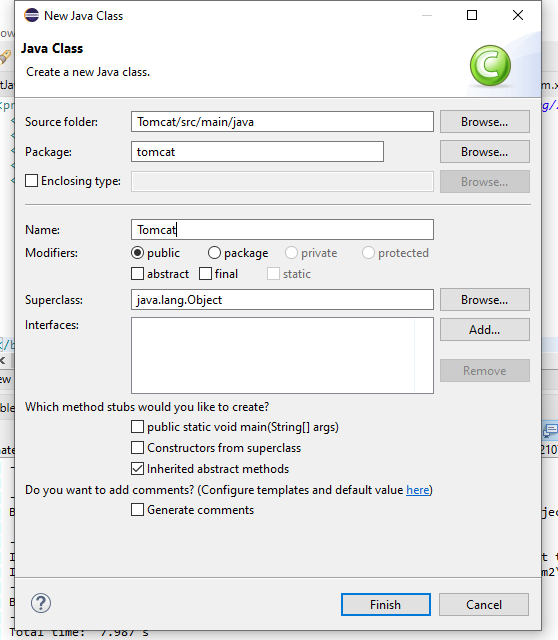


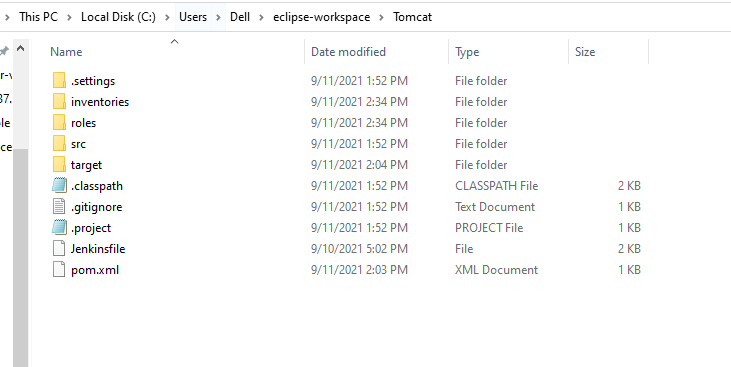


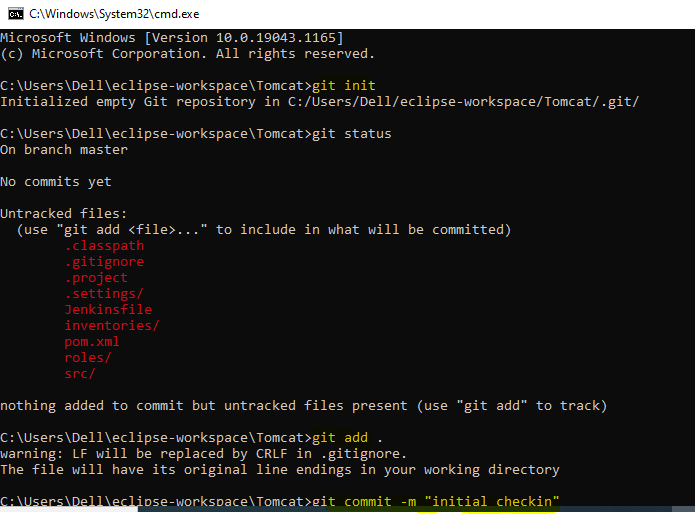


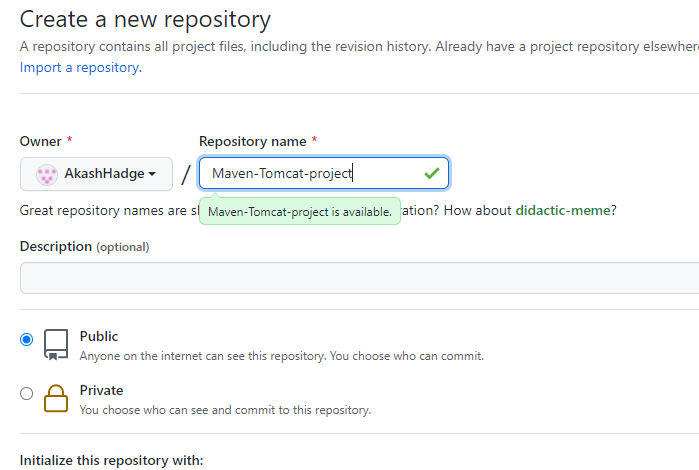


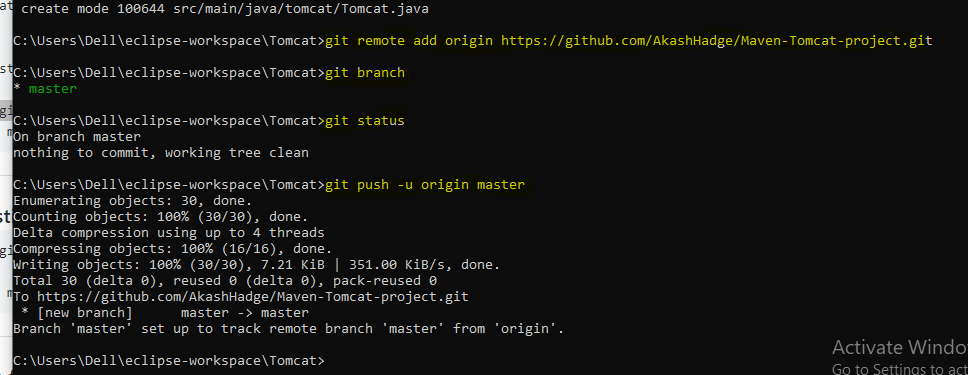


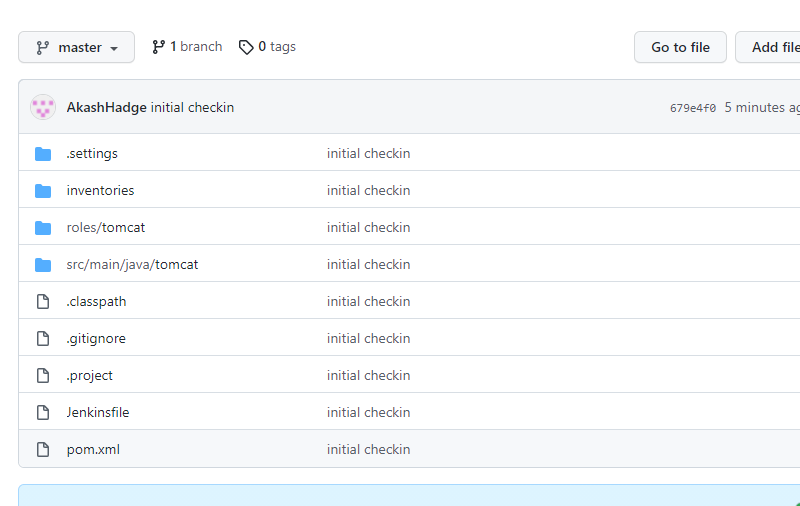


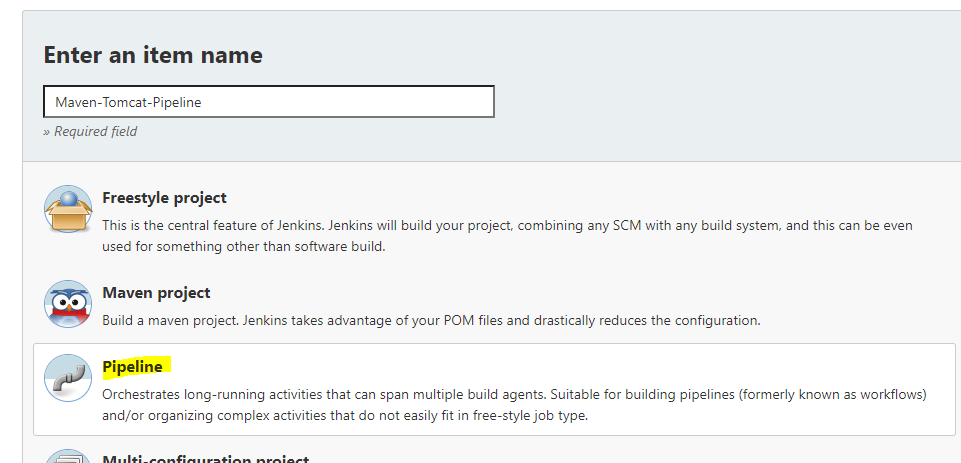


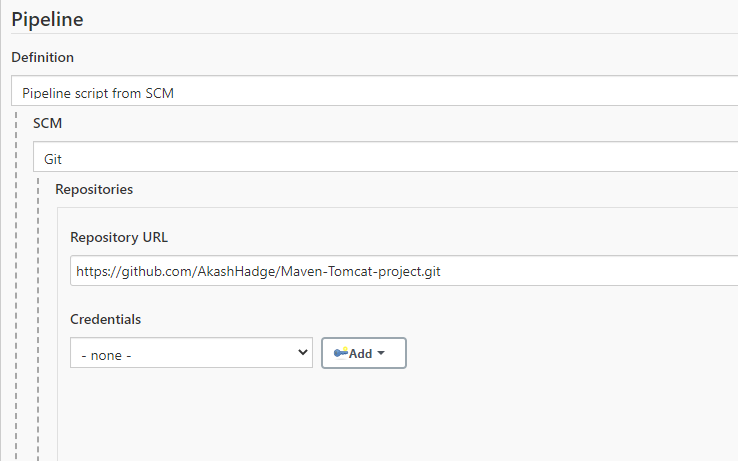


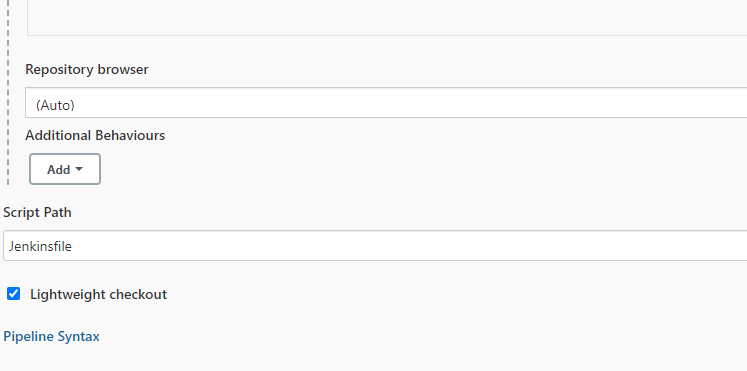


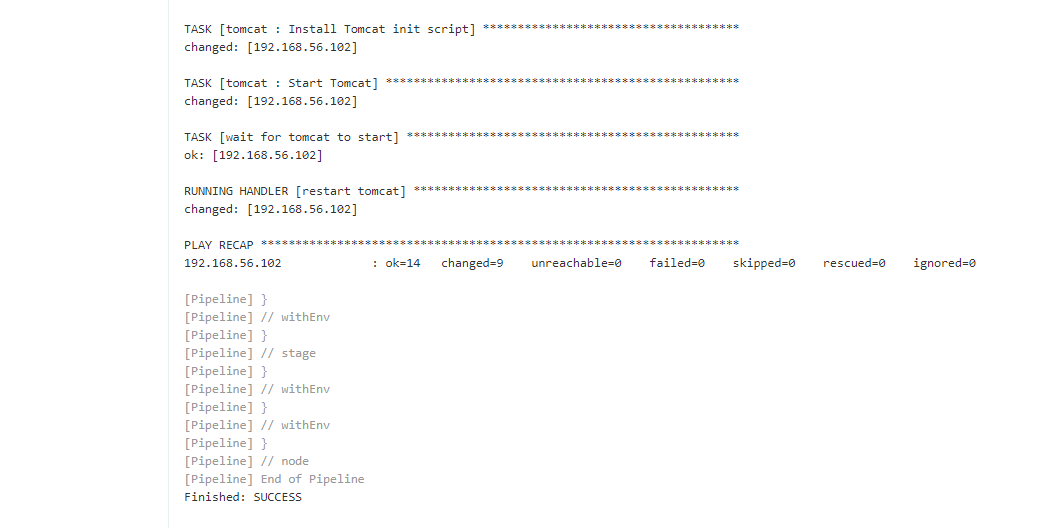


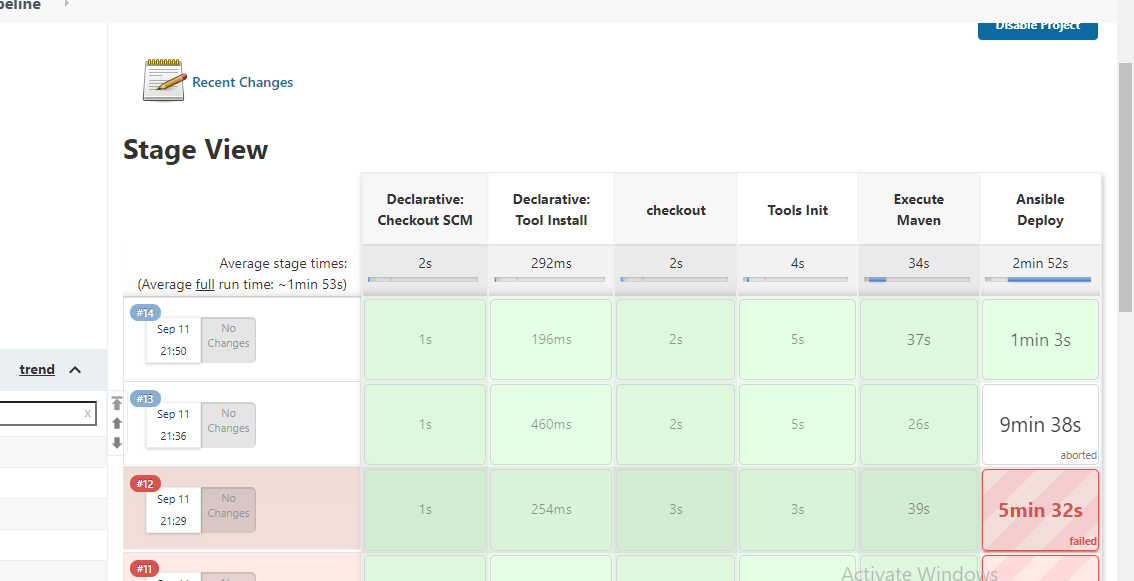


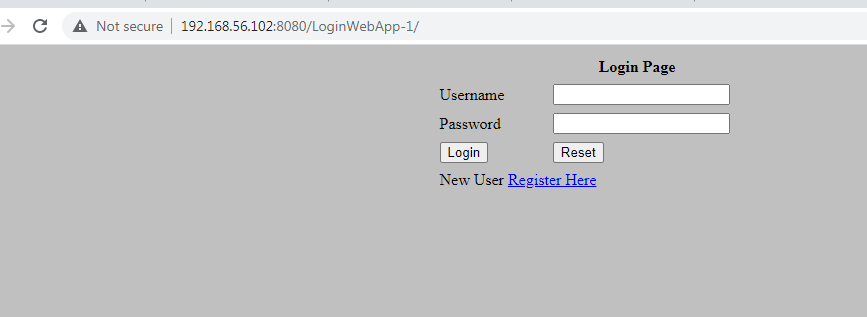


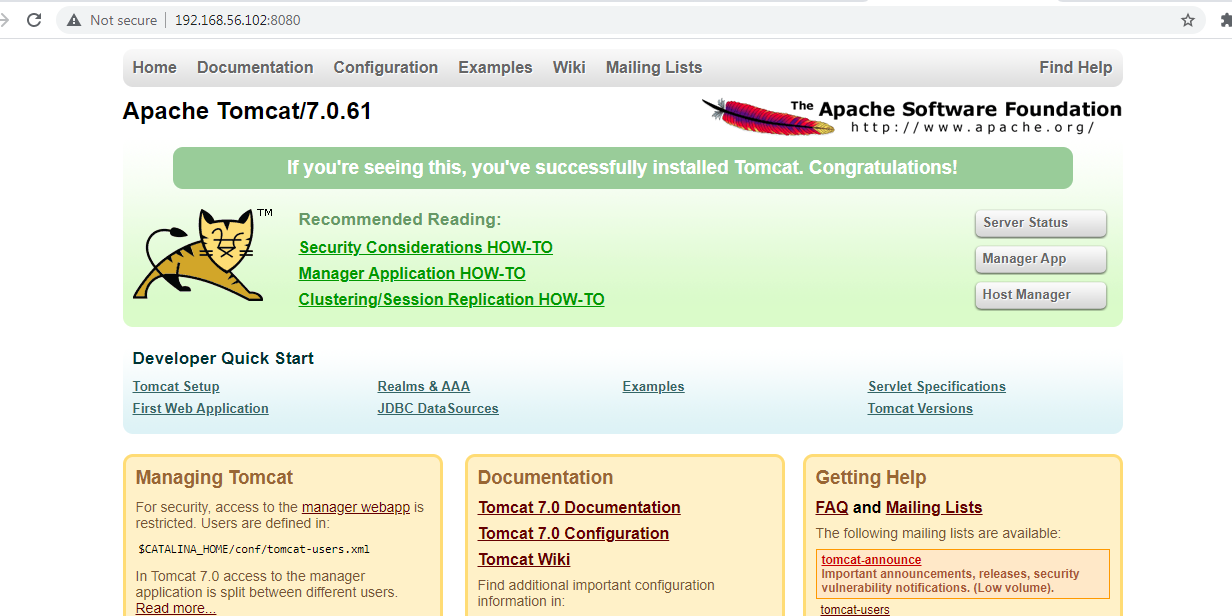












**Assignment-4 Marks: 25**

1. Delete all the existing docker images on your machine.
2. Pull ubuntu 20.04 image from docker hub.
3. Take the backup of this image on your machine and delete this image.
4. Restore the image from backup
5. Run it in an interactive mode and exit.
6. Create a new container and run it in detached mode and login to this container.
7. Create a file “puneet-assessment8.txt” using vi editor inside the container and with the contents "Welcome Puneet to assessment 8".
8. Exit from the container and stop it.
9. Restart the container and login to it.
10. Display the contents of “puneet-assessment8.txt” file.
11. Exit from container.
12. Clean up you machine by deleting all the containers and images.

