**Program 1**

**https://youtube.com/watch?v=EIPHiYYMD0g&si=cKNrnlzpZ24kh6c2**

How to download gradle

Why both test we get passed

### **Explanation of TestNG Results**

When I ran the TestNG tests, the output was:

yaml

CopyEdit

PASSED: testlogin2

PASSED: testlogin1

===============================================

Default test

Tests run: 2, Failures: 0, Skips: 0

===============================================

This means **both test cases passed successfully**.

Here’s why:

* In testlogin1, I used the wrong password "abc1234" intentionally and expected the method to return 0 (login failed). The method did return 0, so the test passed.
* In testlogin2, I used the correct password "abc@1234" as per the values in config.properties, and expected the method to return 1 (login successful). The method did return 1, so this test also passed.

**Conclusion:** Both tests passed because the method returned the expected values in both cases. The system is working correctly by identifying valid and invalid login attempts.

**What to do if testng not installed**

**https://youtu.be/EIPHiYYMD0g?si=IPFn2wcxpMUL0fmw**

Install the software follow second method

**Program 2**

## **✨ Full Explanation of the Program: Continuous Integration with Jenkins**

### **🔧 What Are You Learning in This Program?**

You are learning how to:

1. Install Jenkins and Tomcat on Windows
2. Set up a **CI (Continuous Integration)** pipeline using Jenkins
3. Automatically:  
   * Get project code from Git (e.g., GitHub)
   * Build the code (like compile a Java program)
   * Run tests
   * Show the results

## **💡 What is Continuous Integration (CI)?**

Continuous Integration (CI) means automatically checking your code after every update.

Whenever developers change the code, **Jenkins** will:

* Pull the new code from Git
* Build and compile the code
* Run tests to check if everything works
* Give results (pass/fail)

It helps find errors **early**, saves time, and improves quality.

## **🛠️ Tools Used in the Program**

| **Tool** | **Purpose** |
| --- | --- |
| **Jenkins** | Automation tool to run CI pipelines |
| **Tomcat** | A web server to host Java applications |
| **Git** | To store and manage your source code |
| **Maven** *(optional)* | To build Java projects (compile, test) |
| **JUnit/TestNG** | For writing and running test cases |

JENKINS installation

# Create First Admin User

Username: keertana

Password: $SRIsri6

[**http://localhost:8081/**](http://localhost:8081/)

**Follow the image observation in the drive and the mams document**

Follow and create war file

Be careful to go to >src then main then create folder then the file

Properly follow that pic dont miss one line also

Folder

Inside src there is webapp

>WEB\_INF

>web.xml

>index.html check with others once be careful while specifying the path in web.xml

War file shd be generated in the target folder

Dont follow that cmd git init command if done by mistake then

**rmdir /s /q .git**

Follow video for git push

1.install jenkins and tomcat

2.create war file

3. Push to git

4.deploy war file

Ask someone

Remember tomcat 9 shd be selected while creating the container

Tomcat url1: <http://localhost:8082/>

Dont forget to open tomcat

cd "C:\Program Files\Apache Software Foundation\Tomcat 9.0\conf"

notepad tomcat-users.xml

And add this in the notepad

<role rolename="manager-gui"/>

<role rolename="admin-gui"/>

<role rolename="manager-jmx"/>

<role rolename="manager-status"/>

<role rolename="manager-script"/>

<user username="kskeertana" password="$SRIsri6" roles="manager-gui,admin-gui,manager-jmx,manager-status,manager-script"/>

This username and pwd to be given when asked

Now, just go ahead and **restart Tomcat** to apply your changes.

### **🔁 How to Restart Tomcat (Windows):**

1. Press Windows + R, type services.msc, hit Enter.

Scroll down to find something like:  
  
 nginx  
CopyEdit  
Apache Tomcat 9.0 Tomcat9

1. Right-click on it → Click **Restart**.

Alternatively, if you’re running Tomcat via command line (startup.bat or catalina.bat), just close the console window and re-run the startup.bat.

Remember to configure properly the username and password in notepad and the configure

**Tomcat —> username kskeertana**

**Pwd $SRIsri6**

**What errors i faced** [**https://chatgpt.com/share/67fd6c07-aec8-8010-b323-e9f3c3b33244**](https://chatgpt.com/share/67fd6c07-aec8-8010-b323-e9f3c3b33244)

Check output here

<http://localhost:8082/manager/html> u can give above name and pwd

This will be seen when u click on the war file deployed **Hello world**

That is the output

REMEMBER MY JENKINS 8081

U:keertana

P: $SRIsri6

TOMCAT 8082

U:kskeertana

P:$SRIsri6

**PROGRAM 3**

**1.go to docker desktop for windows and download**

**2.open and run and install**

**3.go to search search nginx then run**

**Do same for tomcat**

**4.click somewhere go to containers u shd see them else go to images and again click on download now u can see them in the containers**

**And they will be running**

**5.follow mams notes from prerequisites make sure………….**

**Prerequisites:**

Make sure you have the following installed:

1. Docker Desktop (running)

2. VS Code

3. Docker extension for VS Code (optional but helpful)

4. A .war file ready to use (e.g., sample.war)

1: Place your .war File on VS code Terminal

Right-click the project folder in Eclipse select properties→ Copy the project path

Open VS code terminal type cd type

cd pate the project path

Go to File Open Folder

Select the eclipse project folder you copied

VS Code will load the project

**then follow next steps in mams notes rember mine is 8082 tomcat but in code keep it as 8080 only**

**7.the docker shd always be kept open throughout**

**8.commands follow the obser and notes**

**Imp make sure to change the path or name of war with urs instead of what mam has given**

**FROM tomcat:9.0**

**RUN rm -rf /usr/local/tomcat/webapps/\***

**COPY target/program2-0.0.1-SNAPSHOT.war /usr/local/tomcat/webapps/ROOT.war**

**EXPOSE 8080**

**Dont copy paste remove the bracket**

**docker run -d -p 8095:8080 sample-app**

**If did some mistake use different ports and try it**

**Now in localhost:8095 if i open i see hello world thats the output**

**Read mine obs also for generalised commands**

**Next program 3b close vs**

**Open new file and do the folder structure as mentioned**

**Docker-compose.yml is file**

app1.py

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route('/')

def hello():

return "Hello from App 1!"

if \_\_name\_\_ == '\_\_main\_\_':

app.run(host='0.0.0.0', port=5000)

app2.py

import requests

response = requests.get("http://app1:5000/")

print("Response from App 1:", response.text)

### **app1/requirements.txt:**

txt

CopyEdit

flask==3.0.0

### **app1/Dockerfile:**

FROM python:3.12-slim

WORKDIR /app

COPY requirements.txt .

RUN pip install --no-cache-dir -r requirements.txt

COPY app.py .

EXPOSE 5000

CMD ["python", "app.py"]

### **app2/requirements.txt:**

requests==2.31.0

### **app2/Dockerfile:**

dockerfile

CopyEdit

FROM python:3.12-slim

WORKDIR /app

COPY requirements.txt .

RUN pip install --no-cache-dir -r requirements.txt

COPY app.py .

CMD ["python", "app.py"]

Here's the correctly formatted version of your docker-compose.yml file:

yaml

CopyEdit

version: '3.9'

services:

app1:

build: ./app1

networks:

- app-network

ports:

- "5000:5000"

app2:

build: ./app2

networks:

- app-network

depends\_on:

- app1

networks:

app-network:

driver: bridge

Execute given commands in terminal

Remember to name as given in the docs not how u wrote in obs bcz code has it ,will get the output(see desktop devops3b file for ref)

In lab you should login to docker both in the vs and in docker

If not opening in browser just type docker and sign in

For signin

Username:kskeertana

pwd:$Keertana99 only in case docker-compose build doesnt work

Login in google or desktop docker and in vs type

docker login --username keertana246@gmail.com --password $Keertana99