D D M M Y Y Y Y
15 02 2025

1.) Explore basic git & github commands

(i) git --version

- This will test the version if its installed
- Make / create a directory in vscode terminal & enter into it :

mkdir cy20

cd cy20

(ii) git config --global username "/name"

(iii) git config --global user.email "email"

- There set the username & email

(iv) git init

To initialize the repository within the folder

(v) git status

This specifies in which branch we are & commit

next create a file with same content, save & execute again

(vi) git add filename

This will add the file & file is indexed

(vii) git commit -m "first commit"

To save this track file in our repository

next add another line of content in the file and save it.

(viii) git branch branch_name

To create a new branch

(ix) git checkout branch-name
To enter a branch
(x) git checkout master
(xi) git fetch
To merge master branch & another branch
(xii) git merge branch_name
(xiii) git log
To know the commit history
(xiv) git branch -d branch_name
To delete branch
(xv) git checkout <commit hash>
To ~~give~~ go to first version of our file
create a new repository in github &
a file
clone this new repo on the local repo
git clone <<https link>>
To push local repo to github acc.
git remote add origin <http...>
git push -u origin matter

D D M M Y Y Y Y
25 02 2025

2) Implement, code, build, test, configure & monitor the software app in Devops using Flask.

```
(i) sudo apt update
(ii) python 3 --version
(iii) sudo apt install python3 -pip -y
(iv) pip --version
(v) sudo apt install python 3- venv -y
(vi) mkdir flask-project
(vii) cd flask-project
(viii) python 3 -m venv flaskenv
(ix) source flask env/bin/activate
(x) pip install flask
(xi) flask --version
(xii) nano app.py
```

```
from flask import Flask
app = Flask (__name__)
@ app.route ('/')
def home ():
    return "Hello, Devops!"
if __name__ == "__main__":
    app.run (host = '0.0.0.0', port = 5000)
```

```
(xiii) python3 app.py
```

(xiv) build:

```
build.sh
#!/bin/bash
echo "setting up environment..."
python3 -m venv flaskenv
source ./venv/bin/activate
pip install flask
echo "Environment setup complete. Run the
app with: Python"
```

(xv) Test:

```
import unittest
from app import app
class TestApp (unittest.TestCase):
        def test_home (self):
            tester = app.test_client()
            response = tester.get ('/')
            self.assertEqual (response.status_code,
            200)
            self.assertEqual (response.data, b"Hello,
            Devops!")
if __name__ == "__main__":
    unittest.main()
```

(xvi) Configure:

```
Create .env file
nano .env
flask-port = 5000
pip install python_dotenv
```

(xvii) monitor :

```
@app.route ('/health')
def health():
    return {"status": "up"}, 200
```

D D M M Y Y Y Y
17 03 2025

3. ~~Imp~~ Install and explore selenium for automated testing.

→
```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import keys
import time
browser = webdriver.firefox()
browser.get ('http://www.yahoo.com')
assert 'Yahoo' in browser.title
elem= browser.find_element (By.NAME, 'p')
elem.send_keys ('selenium' + keys.RETURN)
time.sleep (10)
browser.quit()
```

→
```
from selenium import web driver
from selenium.webdriver.chrome.service import
service
import time
chrome_driver_path ='c:/driver/chromedriver.exe'
service = service (executable_path=chrome_driver_path)
driver = webdriver.Chrome (service = service)
driver.get ("https://www.google.com")
assert 'edge' in driver.title
time.sleep (5)
```

D D M M Y Y Y Y

```
→ import unittest
from selenium import webdriver
class GoogeTestCase (unittest.TestCase):
    def setUp (self):
        self. browser = webdriver. Firefox()
        self. addCleanup • (self. browser. quit)
    def test_page_title (self):
        self. browser.get ("http://www.google.com")
        self. assertIn {"Google', self.browser.title)
    if_name_ == '_main_':
        unittest . main (verbosity = 2)
```

[illegible] 28/3/20

27 03 2025

4. Setting up a gradle project, understanding build scripts (groovy), dependency management and task automation.

* Open source build automation tool used for building java projects, it supports java, groovy and kotlin.
* It automates task like compiling code, running tests, creating tasks
* Open vscode and install
  (i) extension pack for java
  (ii) gradle for java
  (iii) groovy
* Create gradle folder on desktop and open App.java file there and type the code.

```
package gradleproj;
import java.awt.Desktop;
import java.net.URI;
public class App {
    public static void main (String [] args) {
    try {
        String url = "https://www.google.com";
        if (Desktop.isDesktopSupported()) {
            Desktop desktop = Desktop.getDesktop();
            desktop.browse (new URI (url));
        }
```

D.S.C.E.

```
            else {
                System.out.println("Desktop isnt
                supported");
            }
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Using cmd:-

- ★ Download gradle binary bit
- ★ extract it from downloads
- ★ copy gradle-8.13 go to c.drive, program file paste
- ★ go to start, search environment variables.
- ★ select new under sys. var. edit ~~th~~ it & paste the path.
- ★ In cmd ~~paste~~ type :-

```
gradle --version
mkdir gradleproj
cd gradleproj
gradle init
```

- ★ In vscode in build.gradle add at last,

```
task hello {
    dolast {
        println 'Hello, Gradle!' }}
```

save it & go to cmd →

```
gradle task
gradle w hello
```

D.S.C.E.

3/4/25

03/04/2025

5. Implementing continuous security with snyk.

→ Snyk : It is a security tool which is used to find & fix vulnerabilities in the code, dependencies, container image & infrastructure as a code.

① Method - 1
1. Navigate to snyk.io
2. Sign up with GitHub
3. Import the project.
4. Test the code

based severity
(Critical, High, Medium, Low)
factor

② Through command line
1. Install Node js
2. npm install -g snyk
3. snyk auth
4. snyk test or
snyk code test <file-name>

D D M M Y Y Y Y

③ Install snyk in vscode.
→ set up the Authentication of Snyk
→ ~~great~~ $\overset{grant}{\wedge}$ web Acess.
→ download or clone a repo.
→ use snyk tool to check for vulnerabilities and security issue.

8c
17/4/[illegible]

D.S.C.E.

17/04/2025

6. Develop a simple containerized application using Docker.

→ Docker is an open source platform that enables us to build, ship & run applications inside light weight & portable container. It is a tool that is used to package our code, dependencies & run time environment into a single container.

Steps:
- Go to Chrome
- Type docker desktop
- Download docker desktop
- Go to command line
- Check whether wsl is installed or not by typing
  >wsl
- To install
  >wsl --install
- Turn on windows subsystem for Linux
  > In start type 'Turn on windows feature'
  > Select windows subsystem for linux.
  > Restart
- Install the docker fib
  Accept, skip, student, continue
- Click on what is a container & allow.

- https://github.com/docker/welcome-to-docker
- Go to code
- copy the code
- mkdir docker
- cd docker
- git clone <copied code>

```
>cd welcome-to-docker
> docker build -t welcome-to-docker
```

- Go to docker desktop
- click on the IP address

Manually:
- Go to cmd
- mkdir docker_man
- cd docker_man
- notepad app.py

app.py

```
from flask import Flask
app= Flask(_name_)
@app.route('/')
def hello_docker():
    return 'Hello Docker!'
if _name_ == '__main__':
    app.run (host = '0.0.0.0', port = 5000)
```

D D M M Y Y Y Y

<u>requirements.txt</u>
Flask == 2.0.1
Werkzeug == 2.0.1

<u>Dockerfile</u> - all files
Docker Hub
from python : 3.9
container
COPY requirements.txt
RUN pip install --no-cache-dir -r
requirements.txt
container
COPY . .
EXPOSE 5000
CMD ["Python", "app.py"]
all the 3 ~~fill~~ files is docker_man
- Go to cmd

```
> cd docker_man
> ren dockerfile.txt Dockerfile
> docker build -t docker_man
> docker run -p 500:500 docker_man
```

[illegible] 8/5/24

08/05/2025

7. Automate process of running containerised application using Kubernetes.

Kubernetes also called K8S. Its open source platform for automating the deployment scaling & manggement of containerized application.

- Installing using docker desktop
→ In order to install kubernetes we need to install
  Kubectl
  Minikube

→ Go to kubernetes download for windows & scroll down to find install kubectl for windows.

→ Copy the curl command from there

→ Come to cmd & navigate to C drive & make a directory. Enter into that directory and paste that link & hit enter.

→ Follow all the steps & then add the created folders path in environment variables.

→ Next go to install minikube for window
follow all the instructions.
When you type minikube start & suppose
got error, run this command
minikube start --driver=docker

→ After this minikube status.
all parameters must be in running
state this will confirm it is installed
completely

- Before running minikube start, open the
docker desktop

- After minikube status next command

→ Run this → kubectl get nodes

→ After that go to minikube docs &
in that deploy applications cmd
you ~~run~~ in powershell.

Commands
1. Download and run the installer for the
lateral release
→ New-Item-path 'c:\'-Name 'minikube'- Item
type Directory -Force

2. Add the minikube.exe binary to your path
→ $oldPath = [Environment]::GetEnvironmentVariable
('path' [Environment Variable Target]:: Machine)

- if ($oldpath.split(';') -inotcontains 'c:\minikube')
[Environment]:: Set Environment Variable (
'path', $('{0}', c:\minikube' -f $oldpath),
[Environment Variable Target]:: Machine)

3. Start your cluster
→ Minikube restart

4. Interact with your cluster
→ kubectl get po -A
→ Minikube kubectl -get PO -A
→ alias kubectl = "Minikube kubectl" --"
→ Minikube dashboard

5. Deploy applications
→ kubectl create deployment hello-minikube
--image - kibose/echo-server:1.0
→ kubectl expose deployment hello-minikube
-type Nodeport --port = 8080
→ kubectl get services hello-minikube
→ minikube service hello-minikube
→ kubectl port-forward service/hello-
minikube 7080:80

6. Manage your cluster
→ Minikube pause
→ Minikube stop

[illegible]

D D M M Y Y Y Y
15 05 2025

8. Demonstrate creating a job using Jenkins.

→ Jenkins is ~~used~~ a free and open source automation server used for continuous integration & continuous delivery of application to staging production environment.

★ Main features of Jenkins:
- Plugin support: 1800+ plugins for build tools, SCM etc.
- Integration: works with git, docker, kubernetes etc.
- Portability: Runs on windows, linux, macOs etc.
- Extensibility. and concurrent builds.
- Access Control: role based, authentication & authorization.

★ Core tools inside Jenkins include
- Blue Ocean for UI
- Credential manager
- Jenkinfile and pipeline DSL
- SCM integration
- Executor; runs jobs on nodes

★ Uses of Jenkins:
CI/CD automation, code testing and quality checks, dockerimage builds & deployments, Devops orchestration, Release monitoring & roll backs and much more.

D D M M Y Y Y Y

* Steps to install Jenkins software:

1.) Install JDK (latest):

- visit adoptium.net for prebuilt open source JDK libraries.
- Download for windows latest version 21.0.7 (.msi) file.
- Now install setup for JDK
  - → select install on local hardware.
  - → complete installation setup.
- copy the path of bin file
  c:/ program files/ eclipse adoptium/bin

Then go to environment variables setup in the settings, choose path in user variables, add the path & save.

2.) Now to install jenkins software

- Go to jenkins.io and download for windows .msi file
- Now search for local security policies in windows
  - → click on local policies
  - → click on user rights
  - → Search for log on as service
  - → Add user as Administrator (name input) and save & apply ok.

- Install Jenkins
  - → Choose run services as local system already setup as default.
  - → Select test port ex: 8001 random free port
  - → make sure in start service, selected as entire features installed in local hardware
  - → setup installation complete

- Search for services in windows and check if Jenkins in start state & running. can be checked if .out file is running in c:/ program files / jenkins

3.> To run jenkins
- Go to chrome, URL → http:// localhost :8001
- will show up to enter password for administrator
- Copy the generated default password for the session of jenkins in c:/program files/ jenkins/ secrets
- Once you enter the password, ~~not~~ set up this credentials & info of the first admin user.
- Now we have entered into Jenkins software.

4.) To create item/job:

- Click on create a Job.
- click on New item → select freestyle project then click ok
- select build steps in configuration.
  → execute windows batch command
  → echo "projs" %date% %time%
  → save it.
- Now select console output at the navbar left side to see the output of the project that outputs date and time along with its name as in the command saved before.

19/5/

DD MM YYYY
19 05 2025

9. configuration management with Ansible inventory. playbook modules, automating circle configuration with playbooks.

→ Ansible :- It is an open source automation tool used for configuration manage-ment, application deployment, task automation and artistration of multi nodes.

1. Inventory → .ini
2. Modules → reusable units of word → apt
3. Playbooks → YAML (Markup lang.) → .yml
4. Roles → way to organize playbooks and tasks into reusable components.

Steps :-

① Select ~~Ubunto~~ Ubuntu.

② Go to terminal > sudo apt update.

③ > sudo apt install ansible

④ > ansible --version

⑤ > nano hosts.ini

D D M M Y Y Y Y

```
[local]
localhost   ansible_connection = local
```
Save: ctrl + x → Y → Enter

⑥ Playbook : >nano setup.yml
Content ⇒

```
---
-name : Basic Server Setup
 hosts : local
 become : yes
 tasks :
      name: update apt cache
      apt :
         update_cache : yes
      name : Install curl
      apt :
         name : curl
         state : present
```

⑦ Copy the code and open firefox.

⑧ Search yml.validation → paste code & check

⑨ >sudo ansible-playbook -i hosts.ini setup.yml

DD MM YYYY 19 05 2025

10. Creating Maven project, understanding POM file (Project Object Model) POM.xml, dependency management plugins.

→ Maven :- It is an open source build management and project management tool, mainly used for java-based projects. It simplifies the build process, dependency management, reporting and documentation.

* Commands used for Maven:
  - clean → to remove/delete a folder 'target'
  - validate → used to check if the project is correct or not.
  - test
  - compile
  - package → to run a project

POM.xml → Contains metadata, dependencies, plugins, build configurations etc.

Steps:-

① Open Ubuntu.

② Go to the terminal

③ > sudo apt update

④ >sudo apt install openjdk-11-jre-headless -y

⑤ >sudo apt install maven -y

⑥ >mvn --version

⑦ >mkdir lab10

⑧ >cd lab10

⑨ >git --version (if not installed use '>sudo apt install git')

⑩ >git clone https://github.com/krishpluto/Boardgamelistingwebapp.git

>cd BoardgamelistingWebApp

>mvn clean

>mvn validate

>mvn test

>mvn compile

>mvn package

(11) Go to file → BoardGame ... → jar file

(12) > cd target

(13) > java -jar database_service_project-0.0.1.jar

(14) Go to firefox → localhost: 8080