Devops

LAB PRGM-1

-> create a folder -> txt file

-> open the folder in git bash

-> git init

-> git branch -m main (rename the default branch to 'main')

-> git config --local user.email "20cse0140@vvce.ac.in"

or

git config --local user.name "____"

-> git add hello.txt (staging area)

-> git commit -m "commited"

-> git remote add origin "https://....."

if origin already exist -

git remote remove origin

git remote add origin "https://....."

-> git push origin main

-> perform some changes

-> git clone "https:/..."

-> git status

-> git log

-> git reset --hard (Reset the repository to previous state)

clone-new file

pull-existing file

LAB PRGM-2

-> create a folder

-> add txt file

-> git init

-> git add .

-> git commit -m "first commit"

-> git checkout -b "branch1"

-> edit the txt(dont change the entire txt)

-> git add .

-> git commit -m "second commit"

-> git checkout "master"

-> edit -> save

-> git add .

-> git commit -m "third commit"

-> git diff branch1 master

-> git merge branch1

-> change

-> git add .

-> git commit -m "hsdhsde"

-> git merge branch1

LAB PRGM-3

-> create a folder

-> VS code - open the folder created - create a python file

print("Hello world")

-> github - create repository

-> open gitbash in the folder

commands

git init

git config --local user.email "20cse0140@vvce.ac.in"

git add .

git commit -m "commited"

git remote add origin "https://..."

if origin already exist

git remote remove origin

git remote add origin "https://....."

git push origin master

-> open jenkins(browser - localhost:8080 -> username-admin passwords-(progm files->jenkins->.err file->copy the credentials) - 77200362a5234caa9a214b167b3b3812

-> new item

-> name -> frestyle -> ok

-> git project -> paste the url of github repo

-> poll SCM-(* * * * *)

-> save

-> Build now

LAB PRGM-4 - B1

-> create folder

-> vs code -> open folder

-> create a java file - hello.java

-> write a code —> write java code

-> create - Dockerfile

```
FROM openjdk
WORKDIR /app
COPY . /app
RUN javac hello.java
CMD ["java","hello"]
```

```
public class sample {
    public static void main
        (String args[]) {
    System.out.println("Test");
    }
}
```

- -> terminal

javac hello.java

java hello → to check the prog

docker build -t womba .

docker images - to see image.

docker run --name javaapplication javasample womba

user defined

LAB PRGM-5 B2

first cmd

-> docker

-> search -> mysql -> pull

-> open cmd

*first cmd

docker run mysql

docker run -e MYSQL_ROOT_PASSWORD="root123" -d mysql

detach — backend (server model)

*sec cmd → inspect containers.

docker container ls

docker inspect //name//

(copy the ip address)

*third cmd → client model

docker run -it mysql /bin/bash

mysql -h //paste the ip addresss// -u root -p

root123

show databases;

- mysql has built in client & server model
- docker search mysql
- docker pull mysql

→ To stop server container → docker container ls
docker stop //Id of container running//

to check no. of container : docker container ls ; To show No connection run

LAB PRGM-6 : show database;

-> create folder

-> vs code -> open folder

-> create a java file - hello.java

-> write a code

-> create - Dockerfile

FROM openjdk

WORKDIR /app

COPY . /app

RUN javac hello.java

CMD ["java","hello"]

-> terminal

javac hello.java

java hello

docker build -t repository_name .

docker images

docker tag repository_name docker_username/newname:v1.9

(newname-lowercase only)

docker images

docker push docker_username/newname:v1.9

-> dockerhub -> repository -> refresh

LAB PRGM-7 C1 . maven project

-> Open eclipse

-> File -> new -> other -> maven ->maven project -> Next -> next -> all catelog -> maven-archetype-quickstart(org) -> next -> both same(readdatafromjson) -> finish

->// in the console - Y //

readdatafromjson -> right click -> new -> file -> name(student.json) -> finish (get student.json in the list)

{"firstname" : "puneeth",

"lastname" : "shaiva"} - save

-> browser - maven repository

-> search bar - simple JSON

-> click on JSON.simple -> 1.1.1 -> copy the code

-> back to eclipse -> pom.xml -> replace dependency by the copied code -> save

-> readdatafronjson -> src/main -> readdatasort pkg -> right click -> new -> class -> readjson.java

-> paste the code

code

```
package readdatafromjson.readdatafromjson;


import java.io.FileReader;
import java.io.IOException;
import java.text.ParseException;


import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;


public class readjson {
   public static void main(String[] args) throws IOException, ParseException,
org.json.simple.parser.ParseException {
      FileReader reader = new FileReader("student.json");
      JSONParser jsonParser = new JSONParser();
      JSONObject studentObj = (JSONObject) jsonParser.parse(reader);


      String fname = (String) studentObj.get("firstname");
      String lname = (String) studentObj.get("lastname");


      System.out.println("First Name: " + fname);
      System.out.println("Last Name: " + lname);


      reader.close();
   }
}
```

LAB PRGM 8 C2

-> Jenkins -> sign in(user-admin, password-prgmfiles->jenkins->.err->copy credentials)

-> manage jenkins -> plug-ins -> available plug-ins -> search(slack) ->

(if not installed -> installed plug-ins -> right top -> install -> available plug-ins -> search(slack))

-> click on slack -> 2nd link -> type(#devops-demo) -> add jenkins

-> copy redentials (step 3)

-> save settings

-> prev tab

-> manage jenkins

-> system

-> under slack

workspace : vvcegroup

credentials : add -> jenkins -> secret txt(sec drop down) -> paste the credentials (secret) -> add -> drop down(select secreet text) -> save

-> new item -> name -> fresstyle -> ok -> build steps -> execute window batch command(java --version) -> post -> build (slack notifications - check all)

-> save -> build now

LAB PRGM-9 <3

-> browser -> selenium download -> click on java -> save -> extract

-> select browser -> chrome -> documentation

-> older releases -> downloads -> chrome for Testing availability > stable

-> chromedriver win64 -> copy and paste the link in new tab -> save -> extract

open eclipse

(-> File -> new -> other -> Maven -> Maven Project -> next -> next -> maven-archetype-quickstart (select org) -> name for id's (selenium))

(-> selenium -> right click -> build path -> configure build path -> lib -> module path -> add external jars -> selenium (select all except lib) -> open ->apply and close)

-> again right click -> build path -> configure build path -> modulepath -> lib -> select all -> open -> apply and close

different one (seleniumdriver) -> src/main/java

-> selenium -> src -> right click -> new -> class -> Launchchrome -> finish

-> copy paste the code

```
package selenium.selenium;

import java.time.Duration;

import java.util.concurrent.TimeUnit;


import org.openqa.selenium.By;

import org.openqa.selenium.chrome.ChromeDriver;

public class Chromelauncher {

        public static void main(String[] args) throws InterruptedException {

                // TODO Auto-generated method stub

    System.setProperty("webdriver.chrome.driver","C:\\Users\\wwwpr\\Downloads\\chromedriver-win64\\chromedriver.exe");

    ChromeDriver driver = new ChromeDriver();

    driver.manage().window().maximize();

    driver.get("http://www.vvce.ac.in/");

    //Thread.sleep(100);

    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

    driver.findElement(By.xpath("//span[text()='x'][1]")).click();

    //System.out.println(driver.getTitle());

        }

}
```

Launchchrome

you should change

xpath

To (ctrl+f) - to find Xpath in inspect

Convert it into pdf & upload github