



8004

Continuous Assessment for Laboratory / Assignment sessions

Academic Year 2024-25

Name: Aditi Ambasta SAP ID: 60018220113 Course: **DevOps**

Course Code: **DJS22ADC5014** Year: T. Y. B. Tech Sem: V Batch: A1 & A2

Department: Artificial Intelligence(AI) & Data Science

Performance Indicators (Any no. of Indicators) (Maximum 5 marks per indicator)	1	2	3	4	5	6	7	8	A1
Course Outcome	1	2	3	3	4	4	5	5	1,2,3
1. Knowledge (Factual/Conceptual/Procedural/ Metacognitive)	5	5	5	5	5	5	5	5	5
2. Describe (Factual/Conceptual/Procedural/ Metacognitive)	5	5	5	5	5	5	5	5	5
3. Demonstration (Factual/Conceptual/Procedural/ Metacognitive)	5	5	5	5	5	5	5	5	5
4. Strategy (Analyse & / or Evaluate) (Factual/Conceptual/ Procedural/Metacognitive)	4	4	4	4	4	4	4	5	5
5. Interpret/ Develop (Factual/Conceptual/ Procedural/Metacognitive)	-	-	-	-	-	-	-	-	-
6. Attitude towards learning (receiving, attending, responding, valuing, organizing, characterization by value)	4	4	4	4	4	4	4	4	5
7. Non-verbal communication skills/ Behaviour or Behavioural skills (motor skills, hand-eye coordination, gross body movements, finely coordinated body movements speech behaviours)	-	-	-	-	-	-	-	-	-
Total	23	23	23	23	23	23	23	24	25
Signature of the faculty member	Aditi								

Outstanding (5), Excellent (4), Good (3), Fair (2), Needs Improvement (1)

Total Term-work (25) = 24

Sign of the Student:

Aditi

Signature of the Faculty member:
 Name of the Faculty member: Date:

Signature of Head of the Department



EXPERIMENTNO.1

SEMESTER:V

SUBJECT:DevopsLaboratory(DJS22ADL5014)

NAME OF THE STUDENT: Aditi Ambasta ROLLNO.: S004/60018220113

Aim: Write code for a simple user registration form for an event.

To Study DevOps: Principles, Practices, and DevOps Engineer Role and Responsibilities. Post

Lab Question:

What are the Essential Skills required for a DevOps Engineer?

Describe DevOps principle in detail.

DevOps Principles

In short, the main principles of DevOps are automation, continuous delivery, and fast reaction to feedback. You can find a more detailed explanation of DevOps pillars in the CAMS acronym:

Culture represented by human communication, technical processes, and tools

Automation of processes

Measurement of KPIs

Sharing feedback, best practices, and knowledge

Adherence to these principles is achieved through a number of DevOps practices that include continuous delivery, frequent deployments, QA automation, validating ideas as early as possible, and in-team collaboration.

Devops Practices



- Plan – In DevOps planning plays an important role. In this stage, all the requirements of the project and everything regarding the project like time for each stage, cost, etc are discussed. This will help everyone in the team to get a brief idea about the project.
- Code – In this stage the code is written over here according to the client's requirements. Here the code is divided into small codes called Units. This is done to get a clear picture of the code. For example, if the team is doing a project on an online -Ekart application then the login part is divided as one unit, after login the page which shows all the categories is divided as another unit, user profile as another unit, etc.

Some of the examples of the tools used are Git, JIRA

- Build – In this stage building of the units is done. Some of the examples of the tools used are maven, Gradle.
- Test – Testing of all units is done in this stage. So we will get to know where exactly the code is having bugs and if there are mistakes found it is returned. Some of the examples of the tools used are Selenium, PYtest
- Release – At this point, the build is prepared to be deployed in the operational environment. The DevOps department prepares updates or sends several versions to production when the build satisfies all checks based on the organizational demands.
- . An example of the tool used is Jenkins.
- Deploy – In this stage, the code is deployed on the client's environment. Some of the examples of the tools used are AWS, Docker.
- Operate – Operations are performed on the code if required. Some of the examples of the tools used are Kubernetes, open shift.
- Monitor – In this stage monitoring of the application is done over here in the client's environment. Some of the examples of the tools used are Nagios, elastic stack.

DevOps Engineer Roles and Responsibilities

A DevOps Engineer is a link between software development and IT operations. They are



responsible for designing, implementing, and maintaining the software development process and ensuring that the software is deployed efficiently and reliably.

The DevOps Engineer roles and responsibilities may include:

- **Automation:**
DevOps Engineers are responsible for automating as much of the software development process as possible. This includes automating code testing, building, deployment, and monitoring. Automation helps to reduce errors and improve the efficiency of the software development process.
- **Infrastructure as Code:**
DevOps Engineers are responsible for developing and maintaining the infrastructure as code. They use tools like Ansible, Puppet, or Chef to manage infrastructure, and they work with cloud computing platforms like AWS, Azure, or Google Cloud.
- **Continuous Integration and Delivery:**
DevOps Engineers are responsible for implementing and maintaining Continuous Integration and Continuous Delivery (CI/CD) Pipelines. This involves automating the building and testing of code, ensuring that the software is always in a releasable state.
- **Monitoring and Troubleshooting:**
DevOps Engineers are responsible for monitoring the software development process and identifying and resolving issues. They use tools like Prometheus, Grafana, or Datadog to monitor the health of the system, and they work with developers to troubleshoot issues.
- **Collaboration:**
DevOps Engineers work closely with development and operations teams to ensure that the software development process is collaborative and efficient. They communicate with stakeholders and ensure that everyone is aligned with the goals of the project.
- **Security:**
DevOps Engineers are responsible for ensuring that the software development process is secure. This includes implementing security measures throughout the software development lifecycle, conducting security audits, and ensuring that security policies and procedures are followed.

Essential Skills for a DevOps Engineer

DevOps is an approach to software engineering that combines software development with operations, typically to enable faster delivery of software and services. DevOps engineers are responsible for managing the development, deployment, and maintenance of software applications and services. As such, they need to possess a wider range of skills to be successful.



Here are some Essential Skills required for a DevOps Engineer:

- **Automation Skills:**

Puppet, or Chef. They should be able to automate the software development process as much as possible to reduce errors and improve efficiency.

- **Cloud Computing Skills:**

DevOps Engineers should be proficient in cloud computing platforms like AWS, Azure, or Google Cloud. They should have experience with designing and maintaining cloud-based infrastructure, including networking, storage, and security.

- **Containerization Skills:**

DevOps Engineers should be familiar with containerization technologies like Docker (Docker is a platform for developing, packaging, and deploying applications in containers) and Kubernetes (Kubernetes is an open-source container orchestration platform for automating deployment, scaling, and management of containerized applications). They should be able to create and manage container-based deployments and troubleshoot issues.

- **Continuous Integration and Delivery (CI/CD) Skills:**

DevOps Engineers should be experienced with implementing and maintaining CI/CD pipelines. They should have a good understanding of the software development process and be able to automate code testing, building, deployment, and monitoring.

- **Monitoring and Troubleshooting Skills:**

DevOps Engineers should be able to monitor the software development process and identify and resolve issues quickly. They should have experience with monitoring tools like Prometheus, Grafana, or Datadog and be able to troubleshoot issues with developers.

- **Collaboration Skills:**

DevOps Engineers should be able to work collaboratively with development and operations teams. They should have excellent communication and problem-solving skills and be able to foster a culture of collaboration and efficiency.

- **Security Skills:**

DevOps Engineers should be able to ensure that the software development process is secure. They should have experience with implementing security measures throughout the software development lifecycle, conducting security audits, and ensuring that security policies and procedures are followed.

- **Orchestration:**

DevOps Engineers should be able to orchestrate complex, distributed systems to ensure that all components of the system are functioning correctly.

- **Communication Skill:**

DevOps Engineers should possess excellent communication skills. They should be able to collaborate with development and operations teams to identify and resolve issues, and they should be able to communicate the status of the software development process to stakeholders effectively.



CODE:

Frontend:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Register</title>
  <link
  href="https://fonts.googleapis.com/css2?family=Rubik:wght@300;400;500;700&display=swap"
  rel="stylesheet">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">
<style>
:root {
  --bg-dark: #121212;
  --card-dark: #1E1E1E;
  --primary-color: #FF6B6B;
  --text-light: #FFFFFF;
  --text-muted: #B0B0B0;
}

* {
  box-sizing: border-box;
}

body {
  height: 100vh;
  background-color: black;
  background-image:
    linear-gradient(to right, rgba(255, 255, 255, 0.13) 1px, transparent 1px),
    linear-gradient(to bottom, rgba(255, 255, 255, 0.101) 1px, transparent 1px);
  background-size: 80px 80px; /* Size of grid squares */
  color: white;
  font-family: 'Rubik', Arial, sans-serif;
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
  margin: 0;
  padding: 20px;
}
```



}

```
.auth-container {  
  width: 100%;  
  max-width: 500px;  
}  
  
.auth-card {  
  background-color: var(--card-dark);  
  border-radius: 16px;  
  padding: 40px;  
  box-shadow: 0 10px 25px rgba(0, 0, 0, 0.1);  
}  
  
h2 {  
  text-align: center;  
  color: var(--primary-color);  
  margin-bottom: 30px;  
}  
  
.form-group {  
  margin-bottom: 20px;  
}  
  
label {  
  display: block;  
  margin-bottom: 8px;  
  color: var(--text-muted);  
}  
  
input {  
  width: 100%;  
  background-color: var(--bg-dark);  
  border: 1px solid #333;  
  color: var(--text-light);  
  padding: 12px 15px;  
  border-radius: 8px;  
  transition: border-color 0.3s ease;  
  margin-bottom: 10px;  
}
```



```
input:focus {  
    outline: none;  
    border-color: var(--primary-color);  
}  
  
.avatar-preview-container {  
    display: flex;  
    justify-content: center;  
    margin-bottom: 20px;  
}  
  
.avatar-img {  
    width: 120px;  
    height: 120px;  
    border-radius: 50%;  
    object-fit: cover;  
    border: 3px solid var(--primary-color);  
}  
  
.btn-primary {  
    width: 100%;  
    background-color: var(--primary-color);  
    color: var(--bg-dark);  
    border: none;  
    padding: 12px;  
    border-radius: 8px;  
    cursor: pointer;  
    font-weight: 600;  
    transition: background-color 0.3s ease;  
}  
  
.btn-primary:hover {  
    background-color: #FF5252;  
}  
.input-icon {  
    position: relative;  
}  
.input-icon i {  
    position: absolute;  
    left: 15px;  
    top: 40%;
```



```
        transform: translateY(-50%);  
        color: var(--text-muted);  
    }  
    input.icon-input {  
        padding-left: 40px;  
    }  
  </style>  
</head>  
<body>  
  <div class="auth-container">  
    <div class="auth-card">  
      <h2>Register</h2>  
      <form method="POST">  
        <div class="input-icon">  
          <i class="fas fa-user"></i>  
          <input type="text" name="username" id="username" placeholder="Username" class="icon-input" required>  
        </div>  
        <div class="input-icon">  
          <i class="fas fa-envelope"></i>  
          <input type="email" name="email" id="email" placeholder="Email" class="icon-input" required>  
        </div>  
        <div class="input-icon">  
          <i class="fas fa-lock"></i>  
          <input type="password" name="password" id="password" placeholder="Password" class="icon-input" required>  
        </div>  
  
        <div class="form-group avatar-preview-container">  
            
        </div>  
  
        <button type="submit" class="btn-primary">  
          <i class="fas fa-user-plus"></i> Register  
        </button>  
      </form>  
    </div>  
  </div>  
<script>
```



```
document.getElementById('username').addEventListener('input', function() {
  const username = this.value;
  const avatarPreview = document.getElementById('avatar-preview');
  if (username) {
    avatarPreview.src = `https://api.multiavatar.com/${username}.svg`;
  } else {
    avatarPreview.src = `https://api.multiavatar.com/default.svg`;
  }
});
</script>
</body>
</html>
```

BACKEND:

```
from flask import Flask, render_template, request, jsonify
from flask_cors import CORS
from flask_mysqldb import MySQL
from flask_bcrypt import Bcrypt

app = Flask(__name__)
CORS(app)
bcrypt = Bcrypt(app)

# MySQL configuration
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'speakboost_user'
app.config['MYSQL_PASSWORD'] = 'ipd_password'
app.config['MYSQL_DB'] = 'speakboost'

mysql = MySQL(app)

@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']

        # Generate a unique avatar URL using Multiavatar API (using username for uniqueness)
        avatar_url = f"https://api.multiavatar.com/{username}.svg" # Unique avatar URL

        # Hash password
        hashed_password = bcrypt.generate_password_hash(password).decode('utf-8')
```

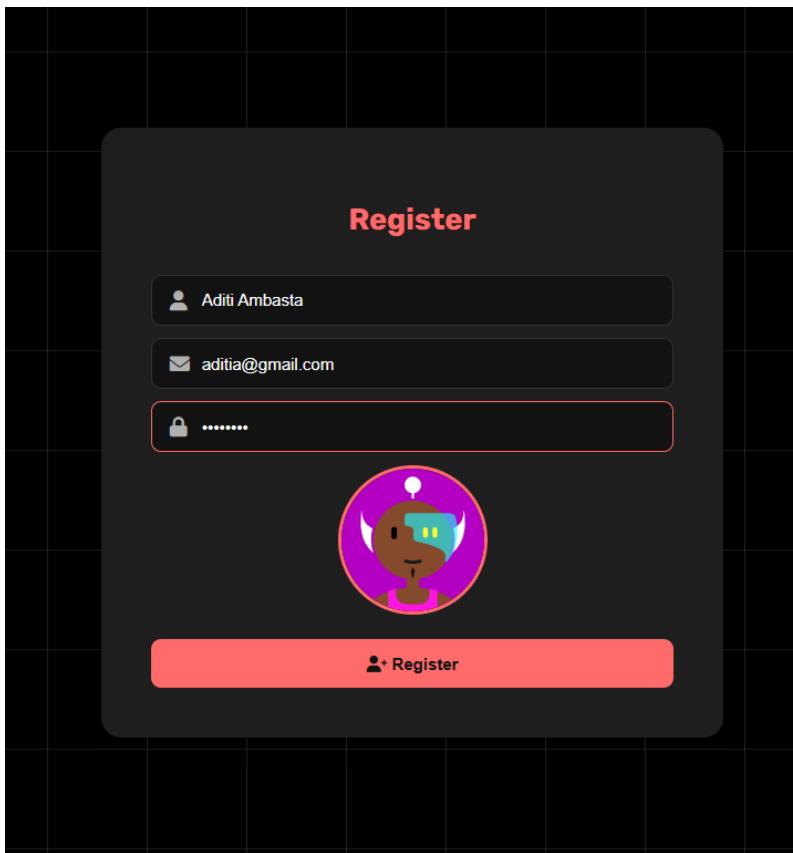


```
# Insert data into the database without role
cur = mysql.connection.cursor()
cur.execute("INSERT INTO users (username, email, password, avatar_url, role) VALUES
(%s, %s, %s, %s, %s)",
            (username, email, hashed_password, avatar_url, None)) # Leave role as None
mysql.connection.commit()
cur.close()

flash("Registration successful! Please log in.", "success")
return redirect(url_for('login')) # Redirect to login page after registration

return render_template('register.html')
```

OUTPUT:





Complete Your Profile

Role:

Student

Age:

23

Interests (comma-separated):

tech

Rate Your Communication Skills (1-10):

8

Save Profile

	id	username	email	password	created_at	role	avatar_url	age	interests	communication_rating
▶	1	aditi	aditi@gmail.com	\$2b\$12\$2v5FCspjJ0xe39YNwQuqaUPnH9XeSh...	2024-11-19 14:02:37	Student	NULL	44	teach	4
2	adi	adi@gmail.com	\$2b\$12\$rlxYt157Ebvrh7ZDKZTen6F260qgBN...	2024-11-19 14:16:06	NULL	NULL	NULL	NULL	NULL	NULL
3	shreya	sh@gmail.com	\$2b\$12\$A7b0STjw0p1oyw4MqJc.kfb34xDY...	2024-11-19 15:03:39	Student	NULL	44	teach	5	NULL
4	new	new@gmail.com	\$2b\$12\$ftemQ4tNxyM1yLn.A95kevW04Tu0...	2024-11-19 16:32:44	Podcaster	https://api.multavatar.com/new.svg	NULL	NULL	NULL	NULL
5	mahi	mahi@gmail.com	\$2b\$12\$w1ueHtHG1H5ZDp0p1QHfc84557lW...	2024-11-19 16:47:16	Podcaster	https://api.multavatar.com/mahi.svg	NULL	NULL	NULL	NULL
6	mira	mira@gmail.com	\$2b\$12\$hpj20GnIEgCBxf3kIAQHeWKXoA7ex...	2024-11-20 20:37:53	Student	https://api.multavatar.com/mira.svg	55	teach	5	NULL
7	mike	mike@gmail.com	\$2b\$12\$aknT/s1Nkvkvn996x12p70kPa2CUNi...	2024-11-21 16:19:26	Student	https://api.multavatar.com/mike.svg	23	technology	6	NULL
8	test	test@gmail.com	\$2b\$12\$/f9OS/ec3jBpJv/pXjnuetuA1RvInqc...	2024-11-22 23:17:20	Social Media Influencer	https://api.multavatar.com/test.svg	23	music	7	NULL
9	Aditi Ambasta	adita@gmail.com	aditia@gmail.com	5.i3ESQ5AD6Re00xFYBBff...	2024-11-26 20:03:24	NULL	https://api.multavatar.com/Aditi Ambasta.svg	NULL	NULL	NULL
●	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Conclusion : Hence we study and understand devops concepts in terms of principle and skill requirement. Also create a single web page using different web based language.

Aditi Ambasta
6001822913
8004 - AIDS

Devops Exp - 01

1. What are the essential skills required for a Devops Engineer

→ Devops is an approach to software engineering that combines software development to enable faster delivery of software & services. Devops Engineer are responsible for managing the development, deployment & maintenance of software applications and services. As such they need to possess a wide range of skills to be successful.

Here are some essential skills:

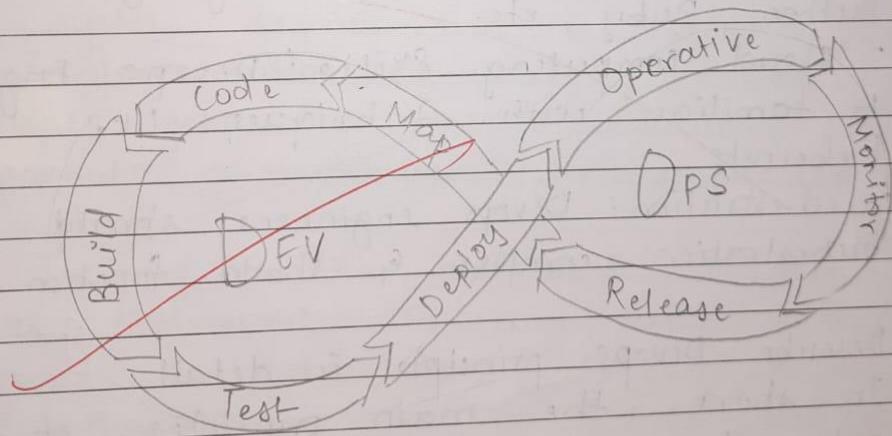
- Automation skills: Devops engineers should have strong automation skills, including experience with scripting language such as Python, Ruby, etc.
- Cloud computing skills: Devops Engineers should be familiar with techniques like Docker & Kubernetes.
- Orchestration: Devops engineers should be able to orchestrate complex & should function correctly.

2. Describe Devops principle in detail

→ In short, the main principles of Devops are automation, continuous delivery & fast reaction to feedback. You can find a more detailed explained of Devops pillars in the CAMS acronym.

- Culture represented by human communication, technical process & tools.
- Automation of process
- Measurement of KPI's
- Sharing feedback, best practices & knowledge

Adherence to these principles is achieved through a number of Devops practices that include continuous delivery, frequent development, deployment, QA automation, validating ideas as early as possible, and in team collaboration.



23/9/24



EXPERIMENT NO. 2

SEMESTER:V

SUBJECT: Devops Laboratory(DJS22ADL5014)

NAME OF THE STUDENT: Aditi Ambasta

SAP ID: S004/60018220113

Aim:To carry out Version Control System / Source Code Management, install git and create a GitHub account.

Theory:

Post Lab Question:

Explain SDLC in details with Diagram

Explain Difference between Git, Github, Github Action, GitLab..(Any three)

Practical:

Prerequisites

➢ Create Username and password for the Github website
(<https://github.com/login>)

Download Git for Windows

1. Browse to the official Git website: <https://git-scm.com/downloads>

2. Click the download link for Windows and allow the download to complete.



Downloads

Mac OS X **Windows** Linux/Unix

Older releases are available and the [Git source repository](#) is on GitHub.

GUI Clients
Git comes with built-in GUI tools (`git-gui`, `gitk`), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

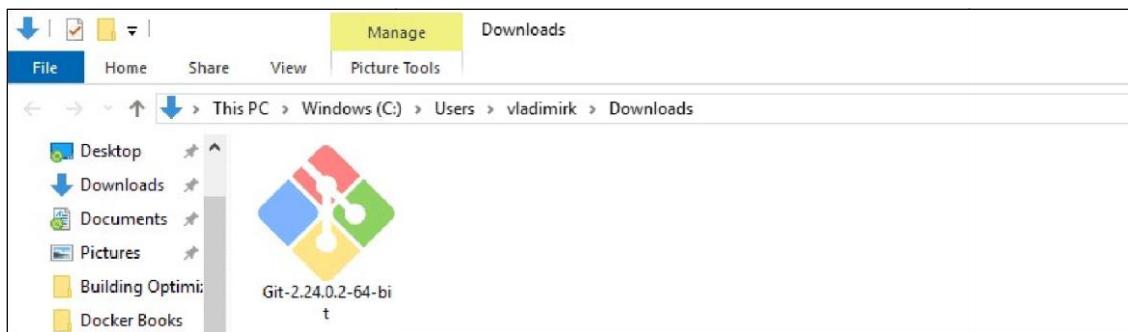
Latest source Release
2.24.0
[Release Notes \(2019-11-04\)](#)
[Download 2.24.0 for Windows](#)

Logos
Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos →](#)

Extract and Launch Git Installer

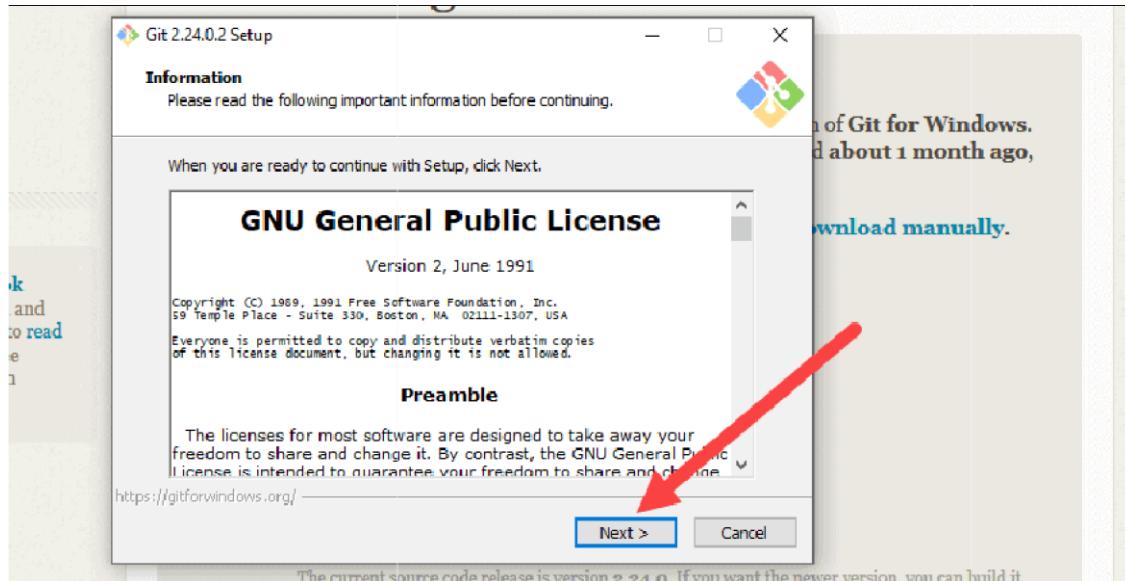
3. Browse to the download location (or use the download shortcut in your browser). Double-click the file to extract and launch the installer.



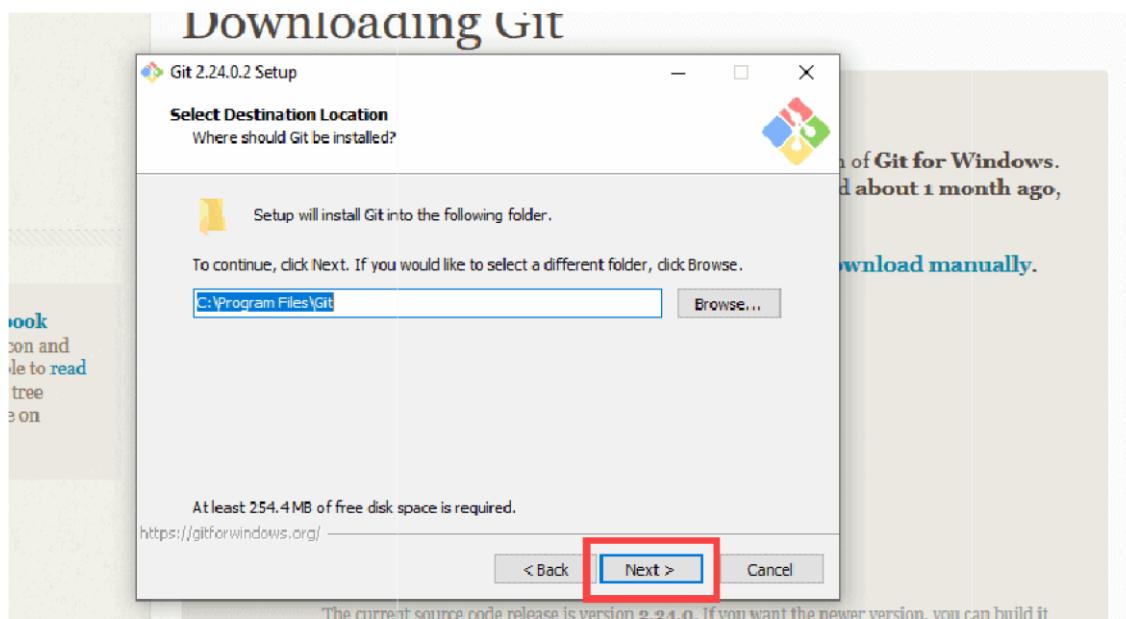
4. Allow the app to make changes to your device by clicking Yes on the User Account Control dialog that opens.



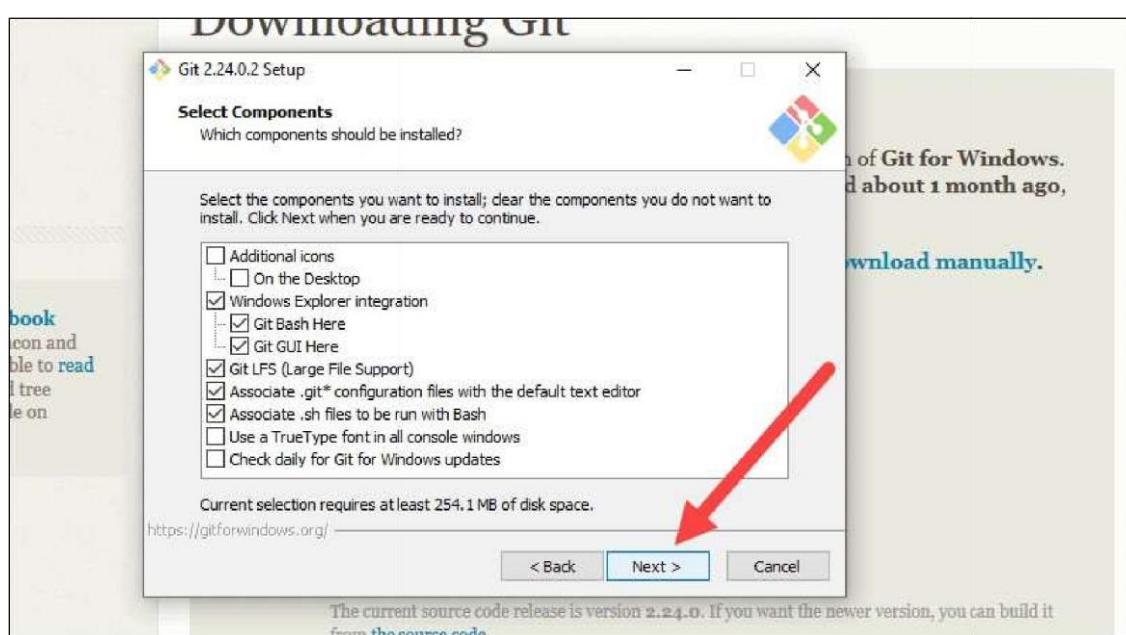
5. Review the GNU General Public License, and when you're ready to install, click Next.

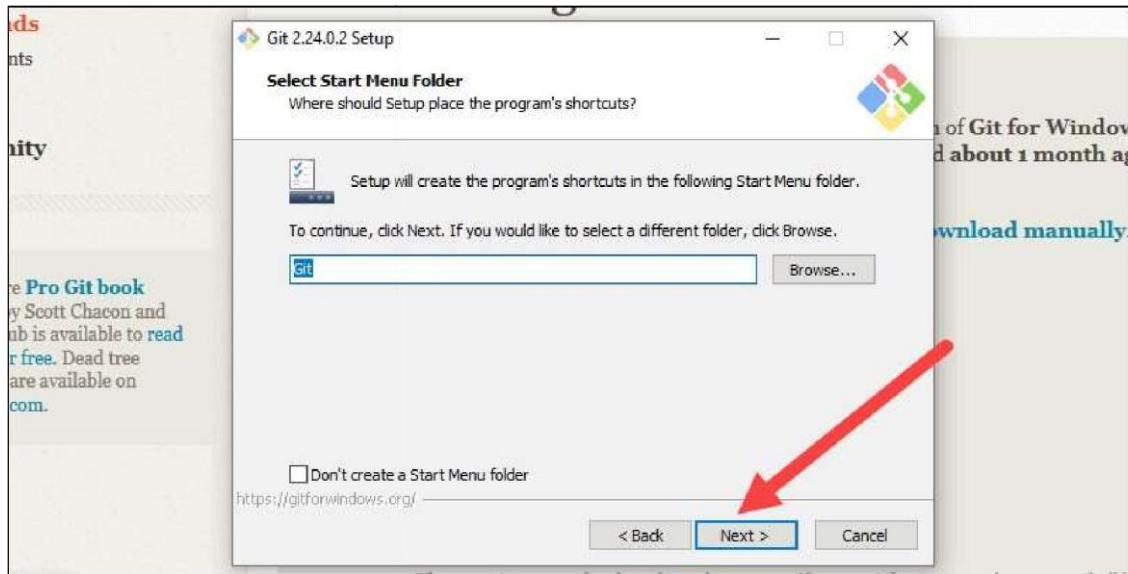


6. The installer will ask you for an installation location. Leave the default, unless you have reason to change it, and click Next.

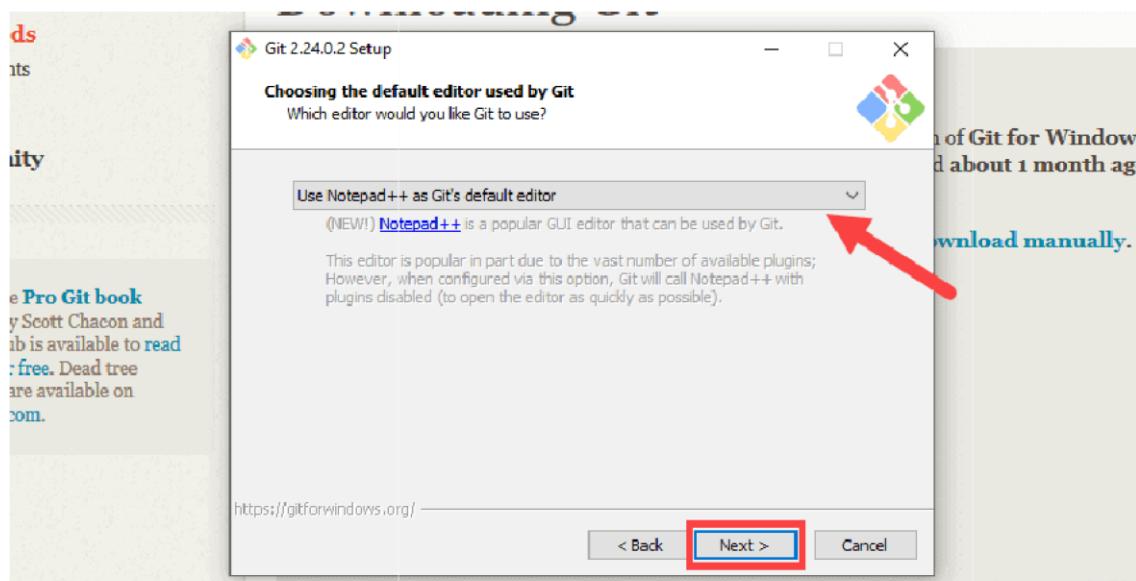


7. A component selection screen will appear. Leave the defaults unless you have a specific need to change them and click Next.





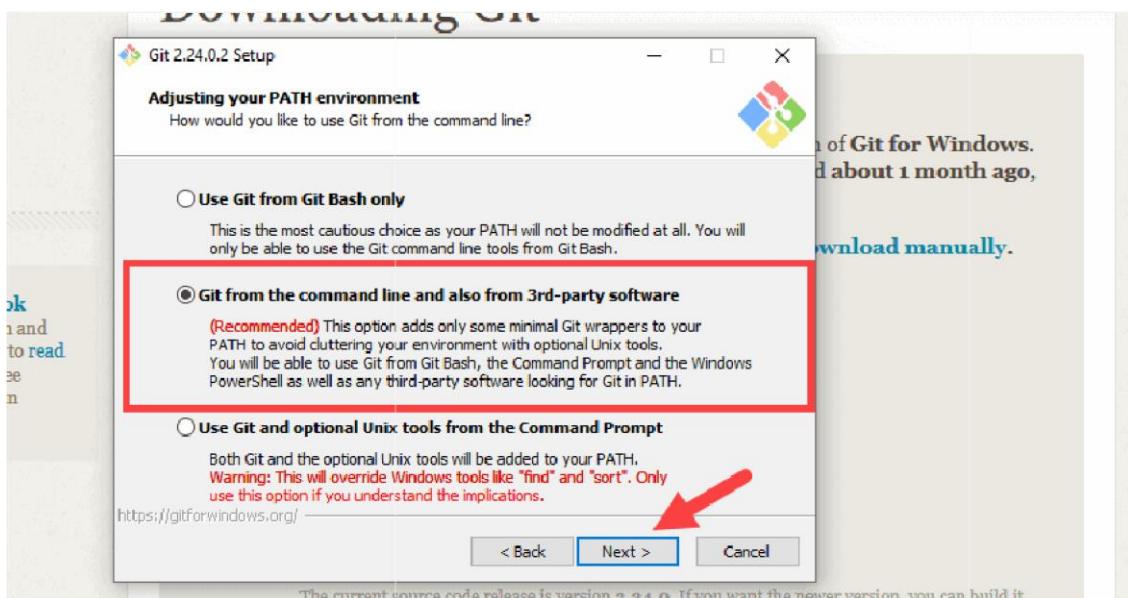
9. Select a text editor you'd like to use with Git. Use the drop-down menu to select Notepad++ (or whichever text editor you prefer) and click Next.



10. The next step allows you to choose a different name for your initial branch. The default is 'master.' Unless you're working in a team that requires a different name, leave the default option and click Next.

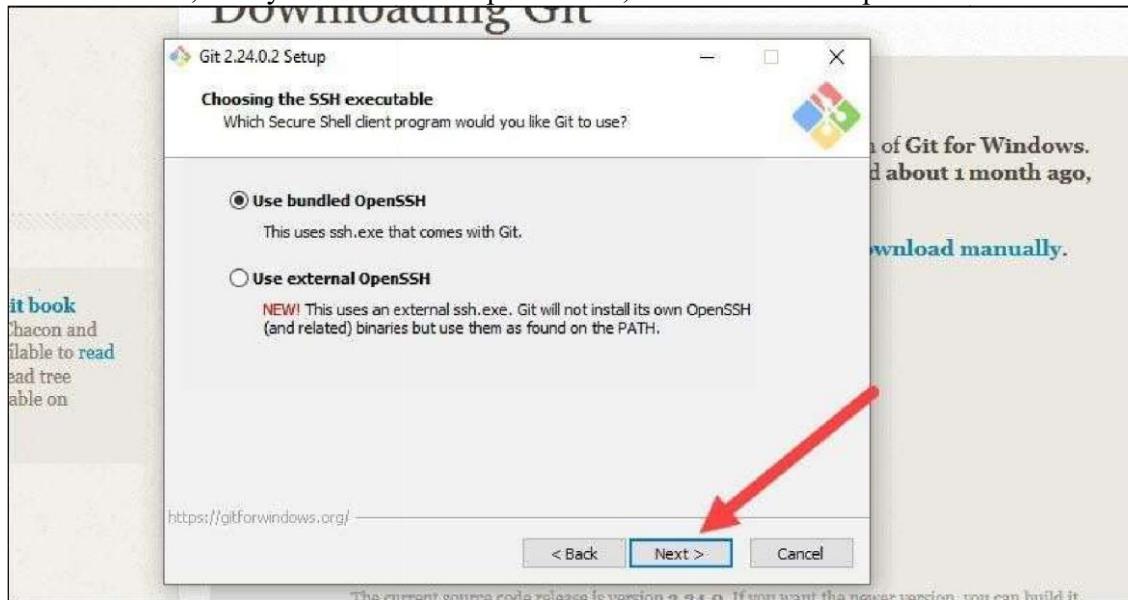


11. This installation step allows you to change the PATH environment. The PATH is the default set of directories included when you run a command from the command line. Leave this on the middle (recommended) selection and click Next.



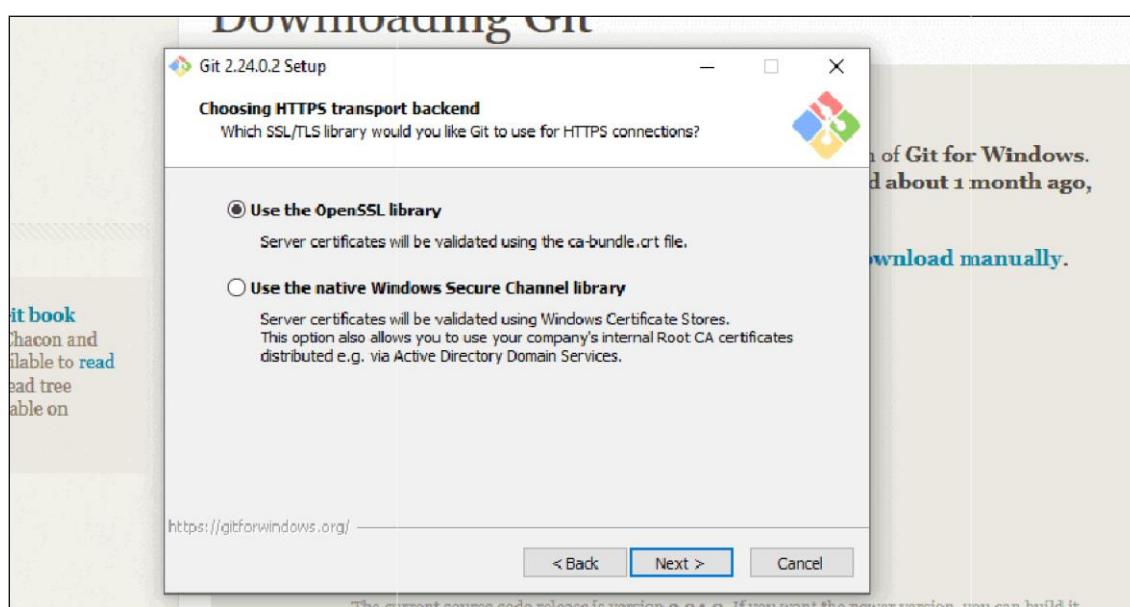


12. The installer now asks which SSH client you want Git to use. Git already comes with its own SSH client, so if you don't need a specific one, leave the default option and click Next.



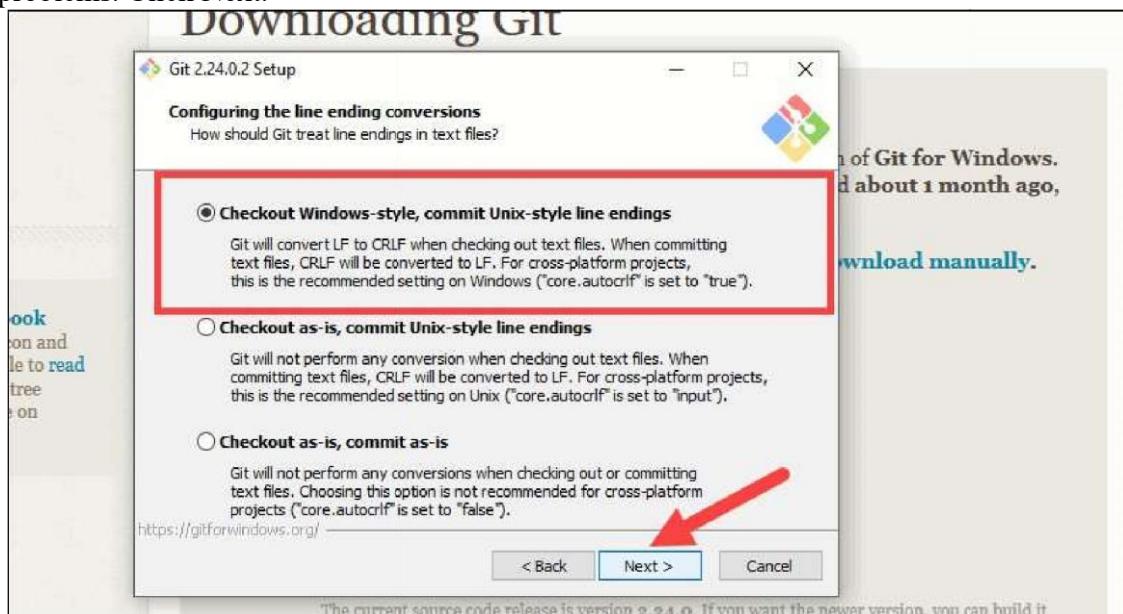
Note: Checkout our comparison of SSH and HTTPS for Git and which one you should use.

13. The next option relates to server certificates. Most users should use the default. If you're working in an Active Directory environment, you may need to switch to Windows Store certificates. Click Next.





14. The next selection converts line endings. It is recommended that you leave the default selection. This relates to the way data is formatted and changing this option may cause problems. Click Next.



15. Choose the terminal emulator you want to use. The default MinTTY is recommended, for its features. Click Next.



entation

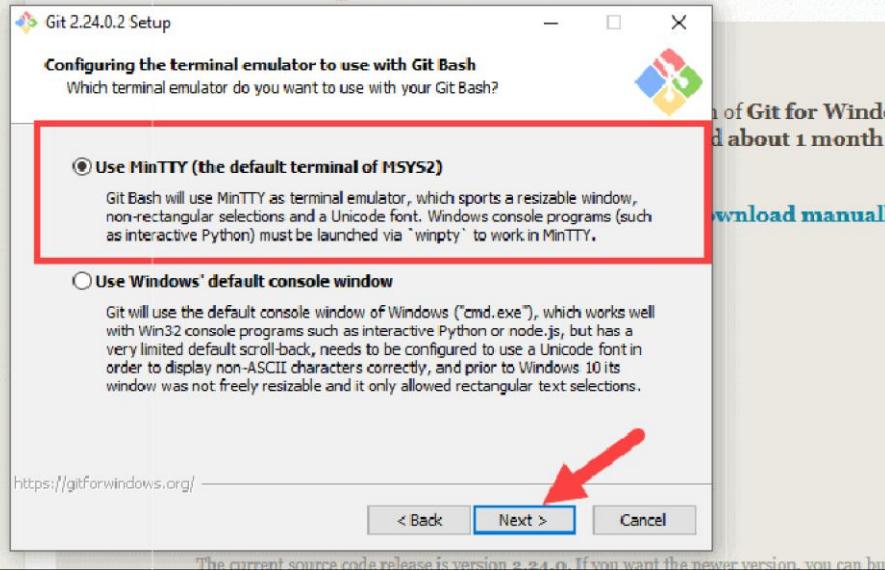
oads

lients

unity

entire **Pro Git** book
n by Scott Chacon and
traub is available to [read
for free](#). Dead tree
ns are available on
[on.com](#).

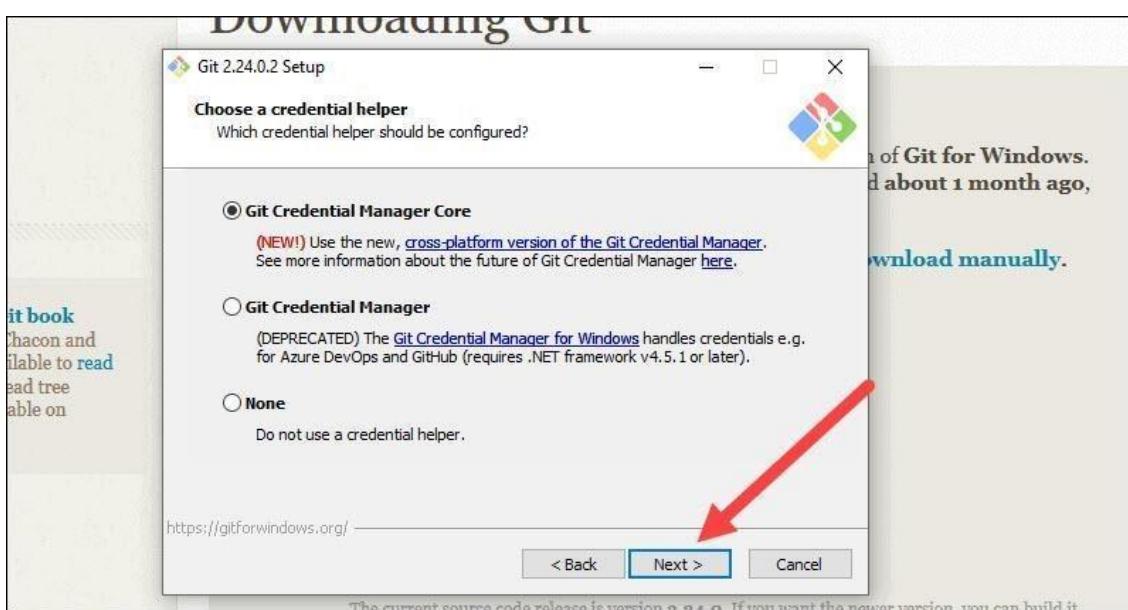
Downloading Git



16. The installer now asks what the git pull command should do. The default option is recommended unless you specifically need to change its behavior. Click Next to continue with the installation.

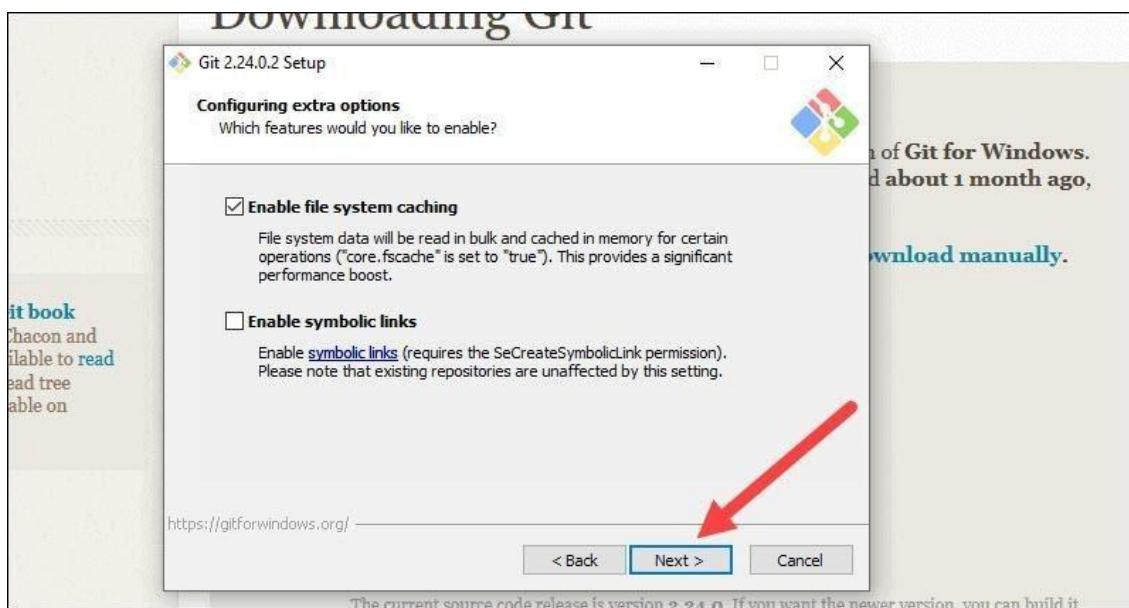


17. Next you should choose which credential helper to use. Git uses credential helpers to fetch or save credentials. Leave the default option as it is the most stable one, and click Next.

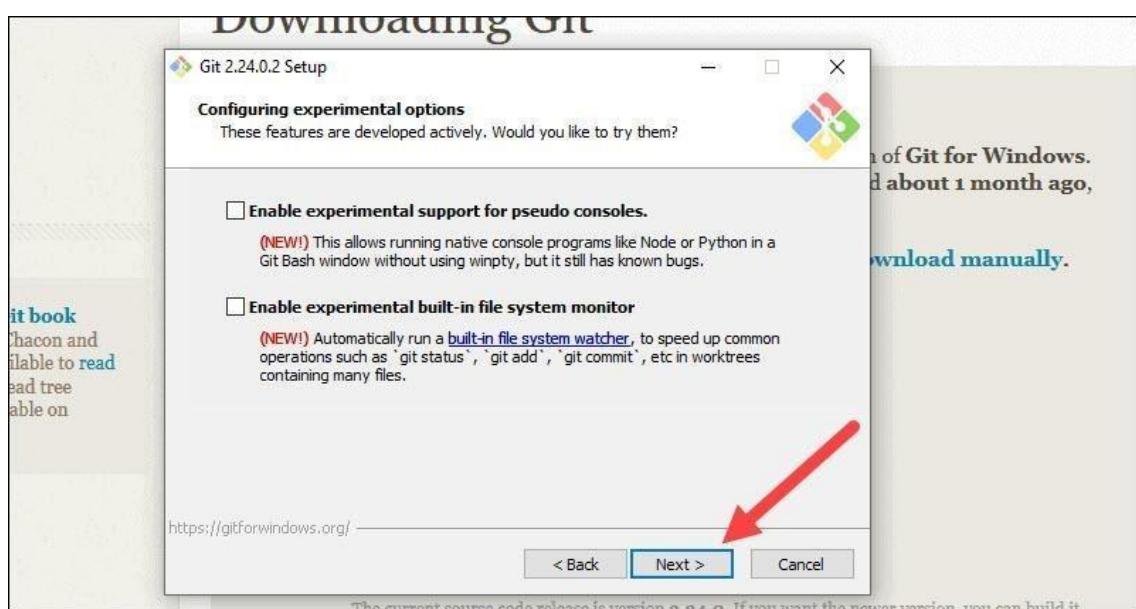


Additional Customization Options

18. The default options are recommended, however this step allows you to decide which extra option you would like to enable. If you use symbolic links, which are like shortcuts for the command line, tick the box. Click Next.

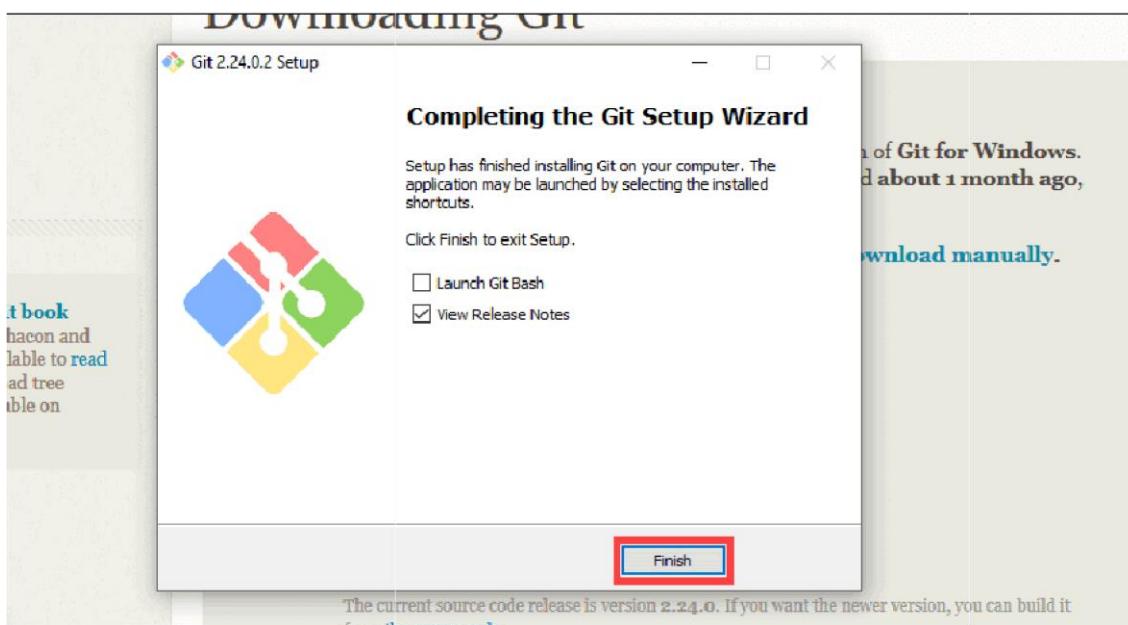


19. Depending on the version of Git you're installing, it may offer to install experimental features. At the time this article was written, the options to include support for pseudo controls and a built-in file system monitor were offered. Unless you are feeling adventurous, leave them unchecked and click Install.



Complete Git Installation Process

20. Once the installation is complete, tick the boxes to view the Release Notes or Launch Git Bash, then click Finish.



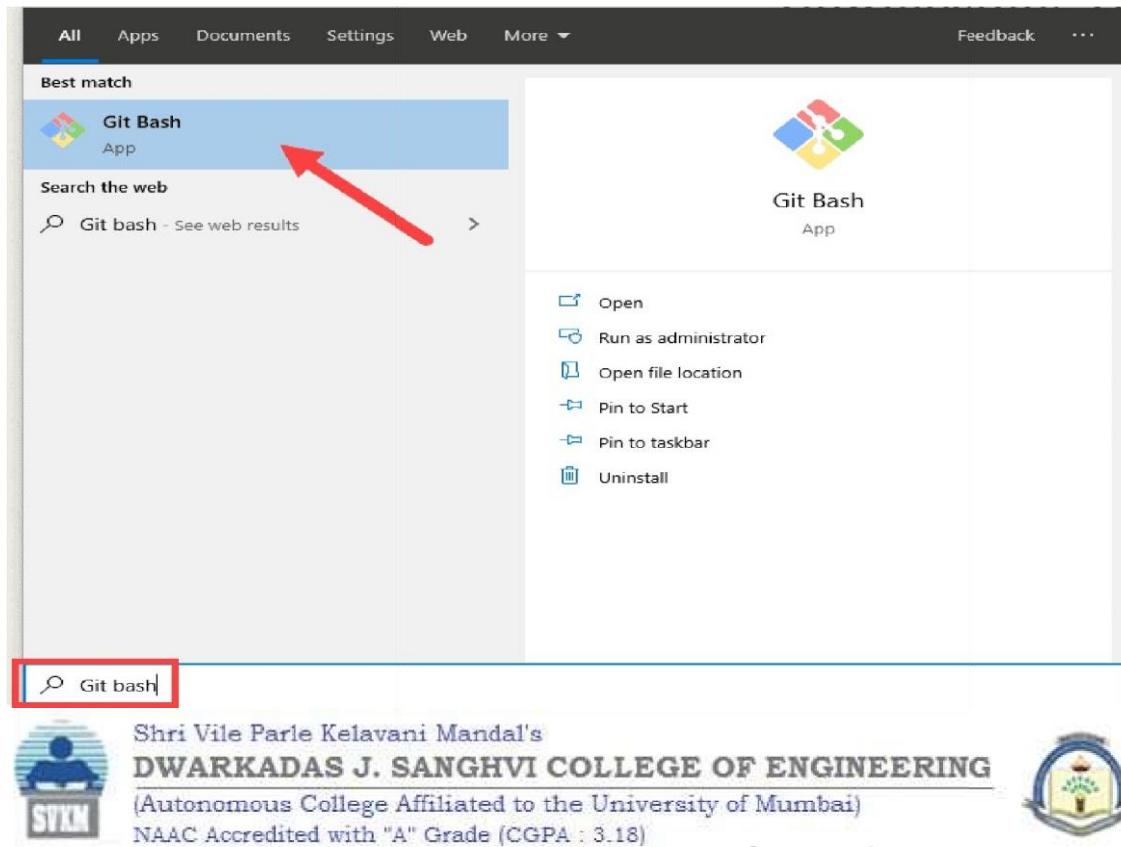
Note: Learn the differences between CLI and GUI. How to

Launch Git in Windows

Git has two modes of use – a bash scripting shell (or command line) and a graphical user interface (GUI).

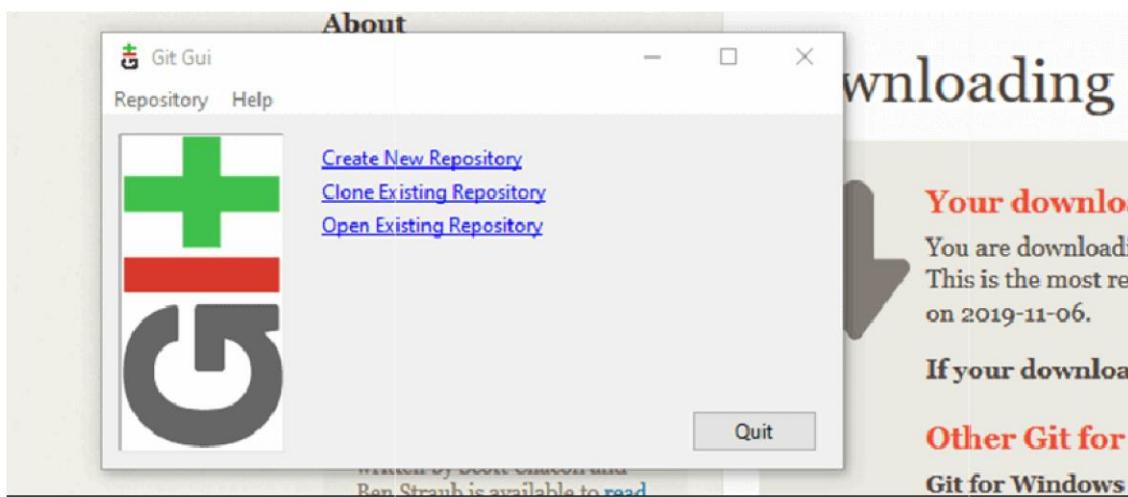
Launch Git Bash Shell

To launch Git Bash open the Windows Start menu, type git bash and press Enter (or click the application icon).



Launch Git GUI

To launch Git GUI open the Windows Start menu, type git gui and press Enter (or click the application icon).



Connecting to a Remote Repository

You need a GitHub username and password for this next step.

Conclusion: hence we practically study the installation of Git and creating GitHub accounts.

Aditi Ambasta
60018220113
S004 - AIE DS

Devops Exp-2

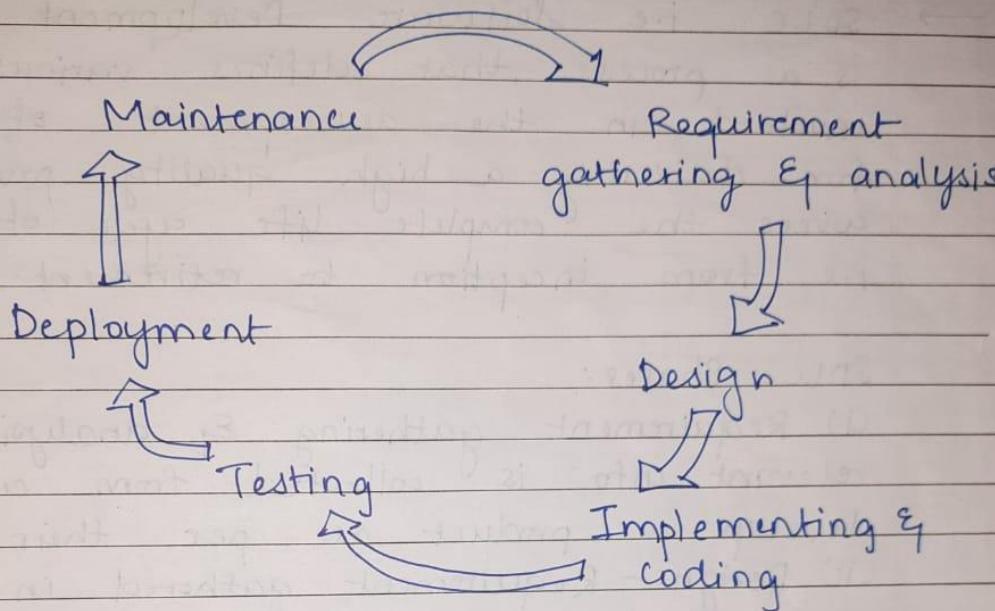
1. Explain SDLC in detail with diagram.

SDLC i.e. Software Development Life Cycle is a process that defines various stages involved in the development of software for delivering a high quality product. SDLC covers the complete life cycle of a software i.e. from inception to retirement.

SDLC Phases:

- (i) Requirement gathering & analysis: - All relevant info is collected from customer to develop a product as per their exception.
- (ii) Design: - Requirement gathered in SRS document is used as an input & software architecture that is used for implementing the system development is observed.
- (iii) Implementation or Coding: - Starts once the developer gets the design document. All the components of software are implemented in this phase.
- (iv) Testing: - The developer software is tested thoroughly & any defects found are assigned to developers to fix them.
- (v) Deployment: - After testing, it is deployed in the production environment or UAT (User acceptance Testing) is done depending on customer expectation.
- (vi) Maintenance: - Maintenance of the product is done i.e. if any issue comes up it

is said to be taken care by the developers.



2. Difference between Git, Github & Gitlab

	Git	Github	Gitlab
(i) Version control system	Cloud Based git	Cloud Based git	Complete Devops
	repo hosting service	repo hosting service	Platform
(ii) Source code management	Repository hosting	Repository hosting	Repo management
	collaboration, CI/CD	collaboration, CI/CD	CI/CD, Devops tools
(iii) Local or remote repository	Cloud based	Cloud based	Self hosted
	repository	repository	repository
(iv) Basis branching & merging	Pull requests, code reviews	Pull requests, code reviews	Merge requests
(v) Command line interface	Web Based GUI	Web Based GUI	Web based GUI
(vi) Issue tracking not included	Integrated issue tracking	Integrated issue tracking	Integrated issue tracking with boards

Conclusion: Thus, we studied about SDLC phases & understand difference b/w GIT, Github & Gitlab



EXPERIMENT NO. 3

Name: ADITI AMBASTA

Roll: S004

SAP:60018220113

Brach: AI & DS

Subject: Devops Laboratory (DJS22ADL5014)

Sem: V

Aim: Perform various GIT operations on local and Remote repositories

Practical:

- Step 1:
To launch Git Bash open the Windows Start menu, type git bash and press Enter (or click the application icon).
 - Step 2:
Check the Git version: `$ git --version`
 - Step 3:
For any help, use the following command: `$ git help config`
provides a manual from the help page for the command just following it (here, it's config).

Another way to use the same command is as follows: `$ git config --help`

- Step 4:
Create a local directory using the following command:
\$ mkdir AIDS
\$ cd AIDS
 - Step 5:
The next step is to initialize the directory:
\$ git init
 - Step 6:
Go to the folder where "AIDS" is created and upload exp.no.1's login page "Loginpage.html"
 - Step 7:
Enter the Git bash interface and type in the following command to check the status:
\$ git status
 - Step 8:
Add the "loginpage" to the current directory using the following command:
\$ git add login.html



- Step 9: (first time user to be followed step.10)
Next, make a commit using the following command:
\$ git commit -m "committing a text file"
- Step 10: if a github account is not available then create a new one.. Link the Git to a [Github](#) Account:

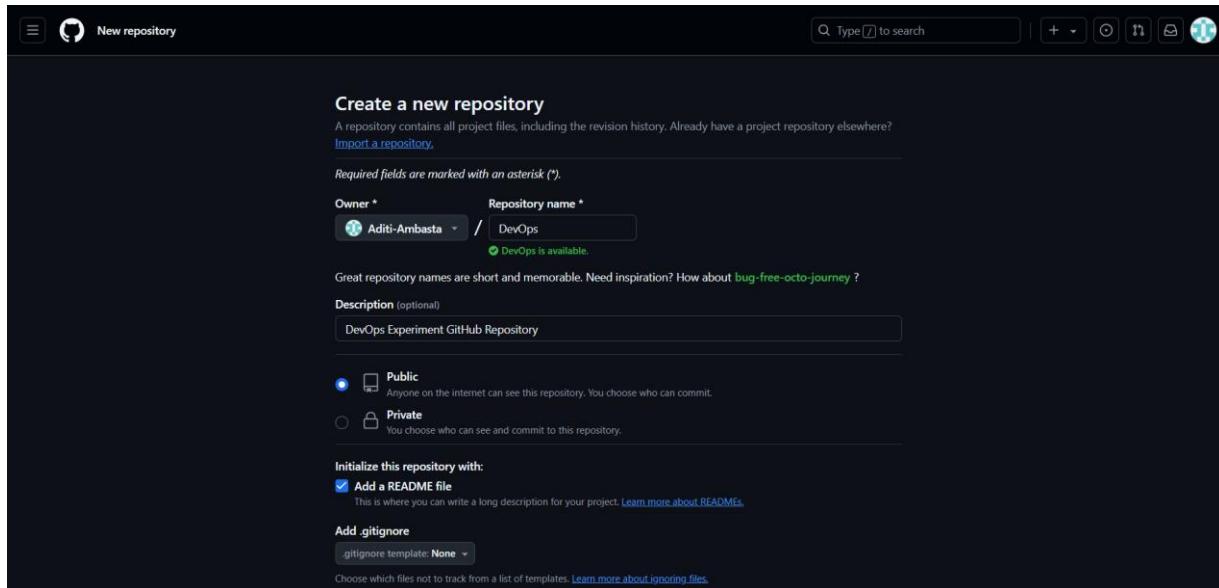
● git config --global user.name "github_username"

● git config --global user.email "email_address"
● Step 11:
Open your Github account and create a new repository with the name "AIDS_demo" or any name and click on "Create repository." This is the remote repository. Next, copy the link of "URL"
- Step 12:
Go back to Git bash and link the remote and local repository using the following command:

```
$ git remote add origin <URL>
```

- Step 13: Push the local file onto the remote repository using the following command:

\$ git push origin master
- Step 14: Move back to Github and click on "AIDS_demo" and check if the local file "Login.html" is pushed to this repository



The screenshot shows the GitHub interface for creating a new repository. The repository is named 'DevOps' and is owned by 'Aditi-Ambasta'. The repository is set to be public. A README file has been added. The repository description is 'DevOps Experiment GitHub Repository'.

Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner * Repository name *
Aditi-Ambasta / DevOps
DevOps is available.

Great repository names are short and memorable. Need inspiration? How about [bug-free-octo-journey](#)?

Description (optional)
DevOps Experiment GitHub Repository

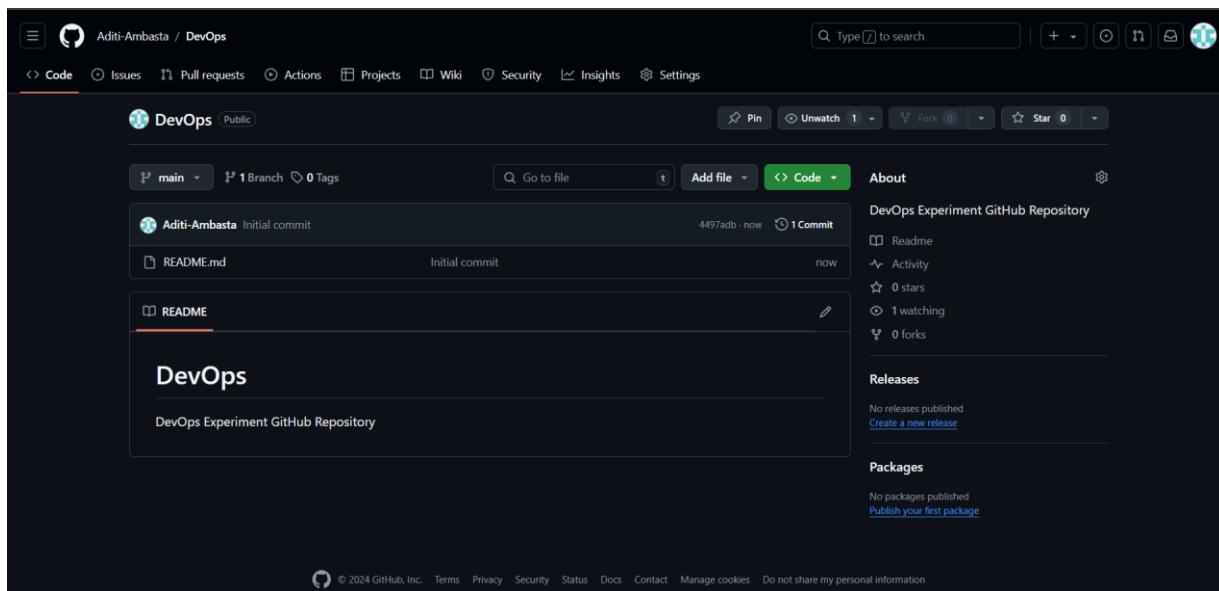
Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:
 Add a README file
This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore
gitignore template: **None**

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).



The screenshot shows the GitHub repository page for 'DevOps'. The repository has 1 commit and 1 branch. The README file contains the text 'DevOps Experiment GitHub Repository'. The repository has 0 stars, 1 watching, and 0 forks. There are no releases or packages published.

Aditi-Ambasta / DevOps

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Code Public

main 1 Branch 0 Tags Go to file Add file About

Aditi-Ambasta Initial commit 4497adb - now 1 Commit

README.md Initial commit now

Code README

DevOps Experiment GitHub Repository

About
DevOps Experiment GitHub Repository
Readme
Activity
0 stars
1 watching
0 forks

Releases
No releases published
Create a new release

Packages
No packages published
Publish your first package

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

Conclusion: Hence we practically studied the installation of Git and creating github accounts.

Aditi Ambasta
60018220113
S004 - A18DS

Devops Exp-03

Aim : To perform various GIT operations on local & remote repositories.

Theory :

Q. Explain the following :

1) Git Add :

Purpose - This command adds changes from your working directory to the staging area (also known as the index). It prepares the files for a commit.

Usage : git add <file>, adds a specific file
git add, adds all changes in the current directory.

Eg:- git add index.html.

2) ~~git commit~~

Purpose : This command records the changes in staging area to the local repo. Each commit is a snapshot of your project at a particular point in time.

Usage : git commit -m "Your message" creates a commit with a message. git commit without -m opens the default text editor to write a commit message.

3) Git push

Purpose: This command uploads your local repo

changes to a remote repository. This typically includes changing sending commits to a branch on the remote.

Usage: `git push origin main` pushes changes to the main branch on the engine remote.

Eg: - `git push origin main`

4) `git pull`

Purpose: This command fetches changes from a remote repository on your local merges into your local branch. It's a combination of `git fetch` & `git merge`.

Usage: `git pull origin main` fetches & merges changes from the main branch on the origin.

Eg: - `git pull origin main`

5) `git clone`

Purpose: This command creates a copy of a remote repository on your local machine.

It also automatically sets up the remote connection, so you can push & pull changes.

Usage: `git clone <repo url>` clones the repo located at given URL

Eg: - `git clone https://github.com/repo.git`

Conclusion: We learnt the commands using git which are crucial for tracking & sharing experimental changes efficiently.

23
23/9/2021

FOR EDUCATIONAL USE



EXPERIMENT NO. 4

Name: ADITI AMBASTA **Roll: S004** **SAP: 60018220113**
Brach: AI & DS **Subject: Devops Laboratory (DJS22ADL5014)** **Sem: V**

Aim: Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to set up a build Job.

Practical:

**Note: Install Java 11 software before installation of jenkins server.

Install Java JDK

Java JDK is required to run Hadoop, so if you haven't installed it, install it.

1. download it from [for example here](#)

2. Run the installation file and the default installation directory will be C:\Program Files\Java\jdk11.2.0\bin

Configure environment variables

1. Open the Start Menu and type in 'environment' and press enter.
 2. A new window with System Properties should open up.
 3. Click the Environment Variables button near the bottom right.

JAVA_HOME environment variable

1. From step 3, find the location of where you installed Java. In this example, the default directory is C:\Program Files\Java\jdk11.2.0
 2. Create a new User variable with the variable name as JAVA_HOME and the value as C:\Program Files\Java\jdk11.2.0\bin

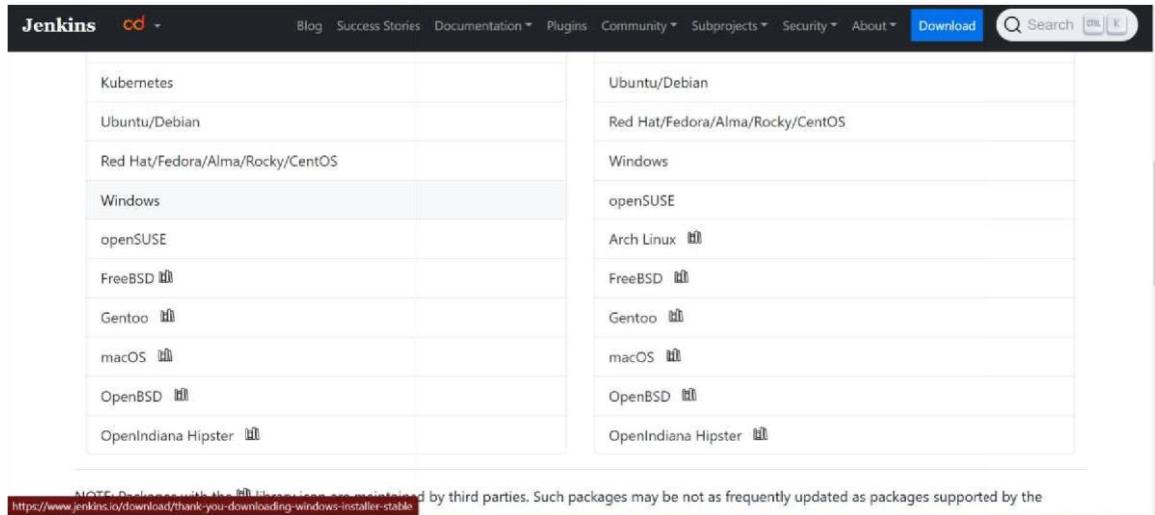
System Variable

1. Select Path → Edit → add C:\Program Files\Java\jdk11.2.0

After installation, open up CMD or Powershell and confirm Java is installed:

```
$ java -version
version "11.2.0"
Java(TM) SE Runtime Environment (build 11.2.0)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)
```

Step1:Go to the URL <https://jenkins.io/download/> to download the windows version or jenkins.war file..



The screenshot shows the Jenkins download page with a grid of operating system distributions and their corresponding Jenkins packages. The grid is as follows:

Operating System	Jenkins Package
Kubernetes	Ubuntu/Debian
Ubuntu/Debian	Red Hat/Fedora/Alma/Rocky/CentOS
Red Hat/Fedora/Alma/Rocky/CentOS	Windows
Windows	openSUSE
openSUSE	Arch Linux
FreeBSD	FreeBSD
Gentoo	Gentoo
macOS	macOS
OpenBSD	OpenBSD
OpenIndiana Hipster	OpenIndiana Hipster

NOTE: Packages with the  icon are maintained by third parties. Such packages may be not as frequently updated as packages supported by the Jenkins community.

<https://www.jenkins.io/download/thank-you-downloading-windows-installer-stable>

Step2:select Download jenkins 2.14.4.2 LTS for:



Downloading Jenkins

Jenkins is distributed as WAR files, native packages, installers, and Docker images. Follow these installation steps:

1. Before downloading, please take a moment to review the [Hardware and Software requirements](#) section of the User Handbook.
2. Select one of the packages below and follow the download instructions.
3. Once a Jenkins package has been downloaded, proceed to the [Installing Jenkins](#) section of the User Handbook.
4. You may also want to verify the package you downloaded. Learn more about verifying Jenkins downloads.

Download Jenkins 2.414.2 LTS for:

Generic Java package (.war)
<small>SHA-256: 922fbf8269fdad1f4bb6540241ed0ce5523a4a532829e155e7bb7fd565b8</small>
Docker
Kubernetes
Ubuntu/Debian
Red Hat/Fedora/Alma/Rocky/CentOS

Download Jenkins 2.428 for:

Generic Java package (.war)
<small>SHA-256: b4f596923eb37fb93c3f5a21a6a32fc3bed57d04a1b63186811c0ce8b3d9f07c</small>
Docker
Ubuntu/Debian
Red Hat/Fedora/Alma/Rocky/CentOS
Windows

Step3: Go to downloaded jenkins.war file and select url enter cmd → open cmd prompt.---> enter command → java -jar jenkins.war



Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)



```
Name           Date modified      Type
C:\Windows\System32\cmd.exe - java -jar jenkins.war
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\jenkins>java -jar jenkins.war
```



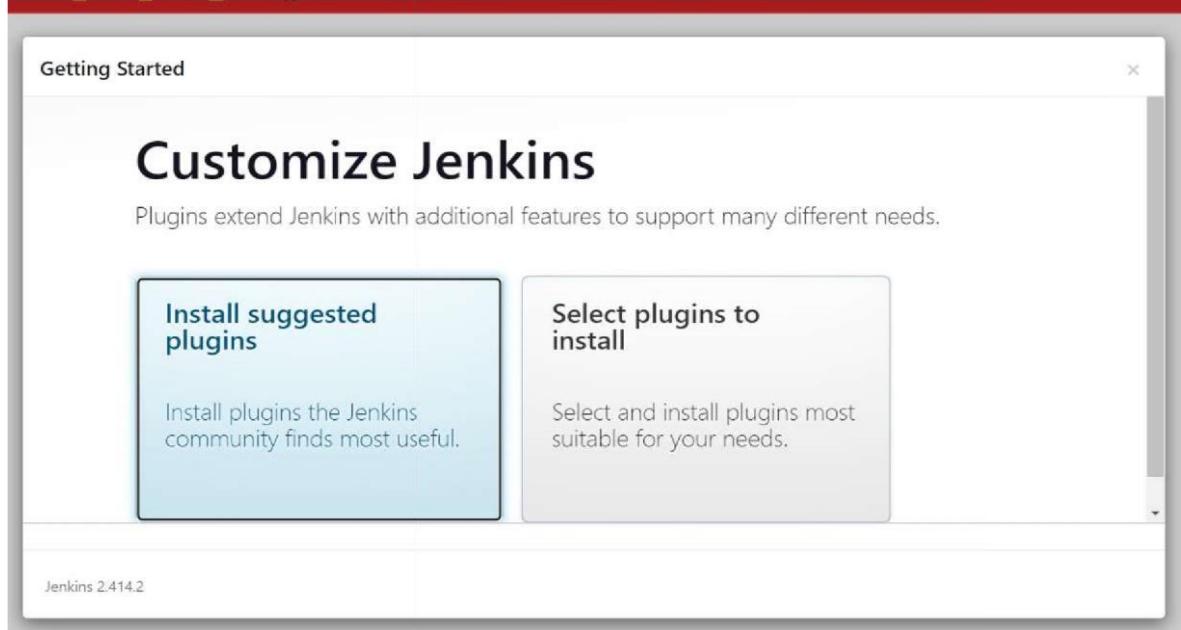
Step4: Jenkins server will start running and it will print password, note down it

```
PS C:\Windows\System32\cmd.exe - java -jar jenkins.war
WARNING: Illegal reflective access by org.codehaus.groovy.vmplugin.v7.Java7$1 (file:/C:/Users/admin/.jenkins/war/WEB-INF/lib/groovy-all-2.4.21.jar) to constructor java.lang.invoke.MethodHandles$Lookup(java.lang.Class,int)
WARNING: Please consider reporting this to the maintainers of org.codehaus.groovy.vmplugin.v7.Java7$1
WARNING: Use -illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2023-10-18 02:04:00.981+0000 [id=33]  INFO  jenkins.install.SetupWizard#init:
*****
*****
*****
Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:
dadae546ed9e49799661ef725f0e2d7f
This may also be found at: C:\Users\admin\.jenkins\secrets\initialAdminPassword
*****
*****
*****
2023-10-18 02:04:31.086+0000 [id=32]  INFO  jenkins.InitReactorRunner$1#onAttained: Completed initialization
2023-10-18 02:04:31.741+0000 [id=58]  INFO  h.m.DownloadService$Downloadable#load: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
2023-10-18 02:04:37.445+0000 [id=58]  INFO  hudson.util.Retry#start: Performed the action check updates server successfully at the attempt #1
2023-10-18 02:04:37.514+0000 [id=24]  INFO  hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
```

Note: password will be available on follow path..

C:\Users\admin\.jenkins\secrets\initialAdminPassword

Step5:Open browser enter url localhost:8080 , Jenkins server page will be open





Step7:enter password which was noted down by you.

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

C:\Users\admin\.jenkins\secrets\initialAdminPassword

Please copy the password from either location and paste it below.

Administrator password

Continue

Step8:It would take time for configuration

Getting Started

Getting Started

Formatter			
<input type="radio"/> Timestamper	<input type="radio"/> Workspace Cleanup	<input type="radio"/> Ant	<input type="radio"/> Gradle
<input type="radio"/> Pipeline	<input type="radio"/> GitHub Branch Source	<input type="radio"/> Pipeline: GitHub Groovy Libraries	<input type="radio"/> Pipeline: Stage View
<input type="radio"/> Git	<input type="radio"/> SSH Build Agents	<input type="radio"/> Matrix Authorization Strategy	<input type="radio"/> PAM Authentication
<input type="radio"/> LDAP	<input type="radio"/> Email Extension	<input type="radio"/> Mailer	

Folders

OWASP Markup Formatter

** Structs
** bouncycastle API
** Instance Identity
** JavaBeans Activation Framework (JAF) API
** JavaMail API

** - required dependency

Jenkins 2.414.2

Step:9 if you want to create a user under admin you enter details otherwise default admin may get selected..



Getting Started

Create First Admin User

Username

Password

Confirm password

Full name

Jenkins 2.414.2

Skip and continue as admin

Save and Continue



Step10:

Getting Started

.....

Confirm password

.....

Full name

dashrath kale

E-mail address

dashrathkal011@gmail.com

Jenkins 2.414.2

[Skip and continue as admin](#)

Save and Continue

Step:11 save and finish

Getting Started

Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.414.2

[Not now](#)

Save and Finish

Step:12



Getting Started

Jenkins is ready!

You have skipped the **setup of an admin user**.

To log in, use the username: "admin" and the administrator password you used to access the setup wizard.

Your Jenkins setup is complete.

[Start using Jenkins](#)

Jenkins 2.414.2

Step:13 Click on Manage jenkins option

The screenshot shows the Jenkins Manage Jenkins page. The sidebar on the left has a 'Manage Jenkins' tab selected. The main content area is titled 'Manage Jenkins'. It features a 'System Configuration' section with links for 'System', 'Tools', 'Nodes', and 'Clouds'. There are also sections for 'Build Queue' (empty), 'Build Executer Status' (2 Idle), and 'Install as Windows Service'. The URL 'localhost:8080/manage' is visible in the address bar.

Step:14 click on available plugin search maven and git and install it...



Jenkins

Dashboard > Manage Jenkins > Plugins

Updates Available plugins Installed plugins Advanced settings Download progress

Plugins

Search available plugins

Install Name Released

<input type="checkbox"/> Command Agent Launcher 107.v773860566e2e	Agent Management	2 mo 11 days ago
<input type="checkbox"/> Oracle Java SE Development Kit Installer 73.vddf737284550	Allows Oracle Java SE Development Kit (JDK) to be installed via download from Oracle's website.	2 mo 15 days ago
<input type="checkbox"/> SSH server 3.312.v1c601b_c83b_0e	Adds SSH server functionality to Jenkins, exposing CLI commands through it.	2 mo 15 days ago

localhost:8080/manage/pluginManager/available

Step:15 Now click on a new item and enter your project name...

Jenkins

Dashboard > All >

Enter an item name

maven build

Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Creates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) organizing complex activities that do not easily fit in free-style job type.



Configure

Source Code Management

General

None

Source Code Management

Git

Build Triggers

Repositories

Build Environment

Repository URL

https://github.com/anujdevopslearn/MavenBuild

Pre Steps

Credentials

- none -

Add

Save

Apply

Build

Post Steps

Build Settings

Post-build Actions

Step:18 enter Goal and option i.e clean compile test finally click on save button

Build

Configure

Root POM

pom.xml

General

Source Code Management

Goals and options

clean compile

Build Triggers

Advanced

Build Environment

Pre Steps

Build

Post Steps

Post Steps

Run only if build succeeds

Build Settings

Post-build Actions

Save

Apply



The screenshot shows the Jenkins dashboard for a 'mavenbuild' project. The left sidebar contains options like 'Build Now', 'Configure', 'Delete Maven project', 'Modules', and 'Rename'. The main area displays a 'Build History' table with one entry: '#1 Oct 18, 2023, 9:43 AM'. Below the table are links for 'Atom feed for all' and 'Atom feed for failures'. The bottom right corner shows 'REST API' and 'Jenkins 2.414.2'.

Step:21 check job's build successfully executed

The screenshot shows the Jenkins console output for build #2. The output includes test results for 'ThingTest', build statistics, and a successful archiving of the build. The log ends with a 'Finished: SUCCESS' message.

```
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.geekcap.vmturbo.ThingTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.166 s -- in com.geekcap.vmturbo.ThingTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] [JENKINS] Recording test results
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time:  46.499 s
[INFO] Finished at: 2023-10-18T09:50:30+05:30
[INFO] -----
[INFO] Waiting for Jenkins to finish collecting data
[INFO] Archiving C:\Users\admin\.jenkins\workspace\mavenbuild\pom.xml to com.java.example/java-example/1.0-SNAPSHOT/java-example-1.0-SNAPSHOT.pom
[INFO] channel stopped
[INFO] Finished: SUCCESS
```

Conclusion : Hence we study continuous integration with help of git ,maven and build one job..

Devops Exp-4

Aim : Continuous Integration

Theory :

1) Explain architecture of CI with every phases & software tools

→ Continuous Integration is a practice in software development where developers frequently integrate their code into a shared repository, typically multiple times a day. The main goal of CI are to detect & fix integration issues early.

Architecture of Continuous Integration

① Build :

Tools :- Jenkins, Travis CI, Circle CI, Bamboo

Phase :- The CI Server triggers the build process after a code change is committed. This step involves preparing the environment & dependencies needed to compile.

② Compile :

Tools :- Maven, Gradle, Ant

Phase :- Source code is compiled into executable code. This step translates the high level code into machine readable format.

③ Code Review

Tools :- Github, Gitlab

Phase :- Before or after the build, code is reviewed by peers or automated tools to ensure it adheres to coding standards.

④ Unit Testing

Tools :- JUNIT, NUNIT, Test NG

Phase :- Automated tests are run on individual components or units of the codebase to ensure they function as expected.

⑤ Integration Testing

Tools :- Selenium, JUNIT, Cucumber

Phase : After unit testing, integration tests are performed to ensure that different components of the application work together correctly.

⑥ Package (WAR, JAR, etc)

Tools : Maven, Gradle, Ant

Phase : Once the code has passed all tests, it is packaged into deployable artifacts such as WAR, JAR or other formats

These artifacts are then stored in an artifact repository for deployment.

Conclusion: CI automates code integration, building, testing & deployment.

23
11/11/2021



EXPERIMENT NO. 5

Name: Aditi Ambasta

Roll: S004

SAP: 60018220113

Brach: AI & DS

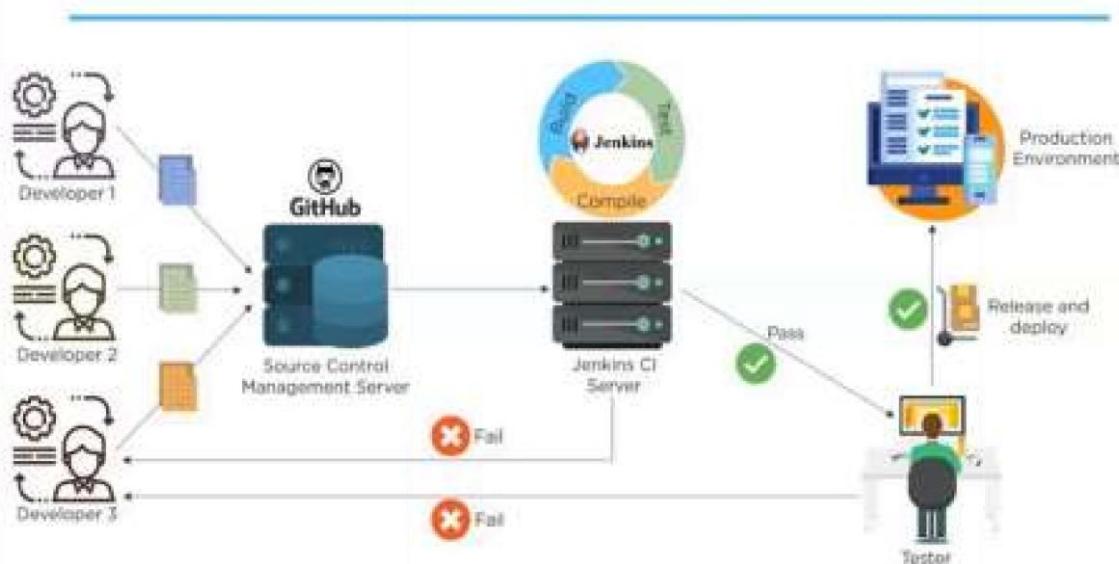
Subject: Devops Laboratory (DJS22ADL5014)

Sem: V

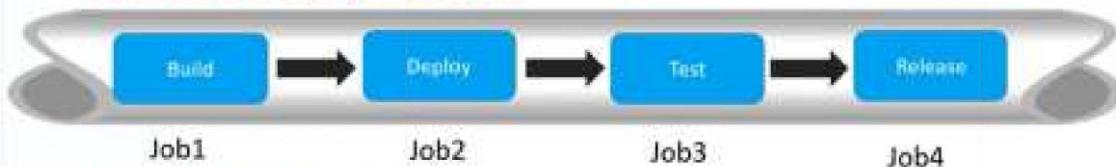
Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins, create a pipeline script to Test and deploy an application.

Practical:

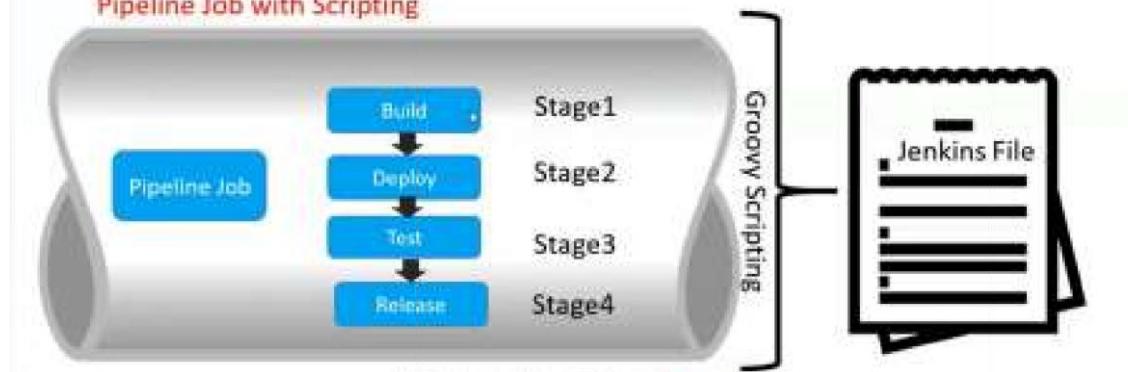
CI & CD



Build And Delivery Pipeline Plugins



Pipeline Job with Scripting





How many Ways we can create Pipeline

- We can Create Jenkins Pipeline in 2 Ways
- 1) Using Build And Delivery Pipeline Plugins
- 2) Using **Groovy Script on the Fly**(Here we use Jenkins file)
 - Scripted
 - Declarative

STEP:1 Click on new item on page

The screenshot shows the Jenkins dashboard with a list of build jobs. The jobs listed are: Build job, Deploy job, First job, Pipe, mavenbuild, new build, new Pipeline, Release job, and Test job. Each job entry includes a status icon, the job name, and some build statistics.

Step 2: Enter Build name you want and select project type ..here we are selecting maven project.. Then click the OK btn.

The screenshot shows the Jenkins 'New Item' search results. The search term 'script pipeline' is entered. The results are as follows:

- Freestyle project**: A description of Jenkins' central feature.
- Maven project**: A description of Jenkins' advantage of using POM files.
- Pipeline**: A description of long-running activities suitable for pipelines.
- Multi-configuration project**: A description of projects with many configurations.

A blue 'OK' button is visible at the bottom of the search results.

Step3: Build trigger → Windows batch command

@echo off date



Dashboard > BuildJob > Configuration

Configure

Build Steps

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Remote Windows batch command

Command

```
See the list of available environment variables.  
echo off  
date  
echo "Build Job"  
exit 0
```

Advanced: +

Add build step +

Save **Apply**

STEP:4 like buildjob create testjob,deployjob,releasejob

Dashboard > Release Job > Configuration

Configure

Build Triggers

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Trigger builds remotely (e.g., from scripts) +

Build after other projects are built +

Projects to watch

Test
TestJob

Trigger only if build is stable

Trigger even if the build is unstable

Trigger even if the build fails

STEP:5 all job are interconnected using manual pipeline

Dashboard > Deploy Job > Configuration

Configure

Build Triggers

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Trigger builds remotely (e.g., from scripts) +

Build after other projects are built +

Projects to watch

Release
Release_Job

Trigger only if build is stable

Trigger even if the build is unstable

Trigger even if the build fails

Save **Apply**

Step 6: Right click on all build open new browser.e.g .. Build job,Test_job , Deploy job.



Job	Last Success	Last Failure	Last Duration
Build_job	N/A	6 days 7 hr #1	7 sec
Deploy_job	N/A	N/A	N/A
First_job	13 days #1	N/A	0.16 sec
hisope	6 days 13 hr #2	6 days 13 hr #1	33 sec
mavenbuild	6 days 7 hr #7	N/A	12 sec

STEP:7 open Build_job first and click on build now button.. It will run the remaining build automatically ...

Upstream Projects

- Build_job
- First_job
- hisope
- mavenbuild

Build History

Build	Time	Status
#1	6 days 13 hr	Pending
#2	6 days 13 hr	Success
#3	6 days 7 hr	Success
#4	6 days 7 hr	Success
#5	6 days 7 hr	Success
#6	6 days 7 hr	Success
#7	6 days 7 hr	Success

STEP:8 Above steps used for manual pipeline.now we want to automate pipeline for that install build pipeline and deliver pipeline plugging from jenkins manage option..

Available plugins

Build Pipeline

Install

Released: 5 yr 10 mo ago

User interface, Build Tests, Other Post-Build Actions

This plugin renders upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins.

Warning: This plugin version may not be safe to use. Please review the following security notices:

- Strong XSS vulnerability

STEP:9 How many times you want to run the pipeline.



Dashboard > Complete pipeline > Configure

Display Options

No Of Displayed Builds

1
2
3
5
10
20
50
100
200
500

No header

Or do not show any column headers

OK Apply

STEP:10 Build Pipeline

Jenkins

Dashboard > Complete pipeline >

Build Pipeline

Trigger a Pipeline Pipeline History Configure Add Step Delete Manage

REST API Jenkins 2.414.2

STEP:11 Select Delivery Pipeline Plugin

Jenkins

Dashboard > Manage Jenkins > Plugins

Plugins

Available plugins

Installed plugins

Advanced settings

Download progress

Search: del

Name	Enabled
Delivery Pipeline Plugin 1.4.2	<input checked="" type="checkbox"/>
This plugin visualizes Delivery Pipelines (jobs with upstream/downstream dependencies)	<input type="checkbox"/>
Pipeline: Declarative 2.2144.0/77a_0f825a_40	<input checked="" type="checkbox"/>
An opinionated, declarative Pipeline.	<input type="checkbox"/>
Report an issue with this plugin	<input type="checkbox"/>
Pipeline: Declarative Extension Points API 2.2144.0/77a_0f825a_40	<input checked="" type="checkbox"/>
APIs for extension points used in Declarative Pipelines.	<input type="checkbox"/>

STEP:12 Delivery the pipeline



Dashboard > New view

1 People
2 Build History
3 Project Relationship
4 Check File Fingerprint
5 Manage Jenkins
6 My Views

Name delivery pipeline

Type

Build Pipeline View
Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the chart.

Delivery Pipeline View
Continuous Delivery pipelines, perfect for visualization on information radiators. Shows one or more delivery pipeline instances, based on traditional Jenkins jobs with upstream/downstream dependencies.

Delivery Pipeline View for Jenkins Pipelines
Continuous Delivery pipelines, perfect for visualization on information radiators. Shows one or more Jenkins Pipeline instances, based on Jenkins pipelines created under this pipeline (0 of 2 available here)

Create

STEP:13 Give the Initial Job name “Build_job”.

Dashboard > delivery pipeline > Pipelines

Components

Component

Name

Please supply a title

Initial Job

Final Job (optional)

OK **Apply**

STEP:14.Same procedure for Delivery Pipeline.

Dashboard > delivery pipeline > Pipelines

Component

Name delivery pipeline

Initial Job

Final Job (optional)

Show upstream

OK **Apply**

STEP 15:



Dashboard > delivery pipeline >

Build History

Edit View

Delete View

Project Relationship

Check File Fingerprint

View Fullscreen

Manage Jenkins

My Views

Build Queue

No builds in the queue.

delivery pipeline

#20 triggered by user admin started 21 minutes ago

```
graph LR; Build_job[Build job] --> Test_job[Test job]; Test_job --> Release_job[Release job]; Release_job --> Deploy_job[Deploy job];
```

Build job
Build job 21 minutes ago - 1 sec

Test job
Test job 21 minutes ago - 1 sec

Release job
Release job 21 minutes ago - 1 sec

Deploy job
Deploy job 21 minutes ago - 1 sec

#19 triggered by user admin started 21 minutes ago

```
graph LR; Build_job[Build job] --> Test_job[Test job]; Test_job --> Release_job[Release job]; Release_job --> Deploy_job[Deploy job];
```

Build job
Build job 21 minutes ago - 1 sec

Test job
Test job 21 minutes ago - 1 sec

Release job
Release job 21 minutes ago - 1 sec

Deploy job
Deploy job 21 minutes ago - 1 sec

#18 triggered by user admin started 23 minutes ago

```
graph LR; Build_job[Build job] --> Test_job[Test job]; Test_job --> Release_job[Release job]; Release_job --> Deploy_job[Deploy job];
```

Build job
Build job 23 minutes ago - 1 sec

Test job
Test job 23 minutes ago - 1 sec

Release job
Release job 23 minutes ago - 1 sec

Deploy job
Deploy job 23 minutes ago - 1 sec

STEP 16:

delivery pipeline

#20 triggered by user admin started 21 minutes ago

```
graph LR; Build_job[Build job] --> Test_job[Test job]; Test_job --> Release_job[Release job]; Release_job --> Deploy_job[Deploy job];
```

Build job
Build job 21 minutes ago - 1 sec

Test job
Test job 21 minutes ago - 1 sec

Release job
Release job 21 minutes ago - 1 sec

Deploy job
Deploy job 21 minutes ago - 1 sec

#19 triggered by user admin started 21 minutes ago

```
graph LR; Build_job[Build job] --> Test_job[Test job]; Test_job --> Release_job[Release job]; Release_job --> Deploy_job[Deploy job];
```

Build job
Build job 21 minutes ago - 1 sec

Test job
Test job 21 minutes ago - 1 sec

Release job
Release job 21 minutes ago - 1 sec

Deploy job
Deploy job 21 minutes ago - 1 sec

#18 triggered by user admin started 24 minutes ago

```
graph LR; Build_job[Build job] --> Test_job[Test job]; Test_job --> Release_job[Release job]; Release_job --> Deploy_job[Deploy job];
```

Build job
Build job 24 minutes ago - 1 sec

Test job
Test job 24 minutes ago - 1 sec

Release job
Release job 24 minutes ago - 1 sec

Deploy job
Deploy job 23 minutes ago - 1 sec

STEP 17:

Pipeline concepts

- Pipeline**
- A Pipeline is a user-defined model of a CD pipeline. A Pipeline's code defines your entire build process, which typically includes stages for building an application, testing it and then delivering it.
- Also, a **pipeline block** is a key part of **Declarative Pipeline syntax**.
- Node**
- A node is a machine which is part of the Jenkins environment and is capable of executing a Pipeline.
- Also, a **node block** is a key part of **Scripted Pipeline syntax**.
- Stage**
- A stage block defines a conceptually distinct subset of tasks performed through the entire Pipeline (e.g. "Build", "Test" and "Deploy" stages), which is used by many plugins to visualize or present Jenkins Pipeline status/progress.
- Step**
- A single task. Fundamentally, a step tells Jenkins what to do at a particular point in time (or "step" in the process). For example, to execute the shell command make use the sh step: sh 'make'. When a plugin extends the Pipeline DSL, that typically means the plugin has implemented a new step.

OR

Script pipeline method two Method

1. Scripted

Step1: create project using Pipeline project



2. Pipeline option → Pipeline

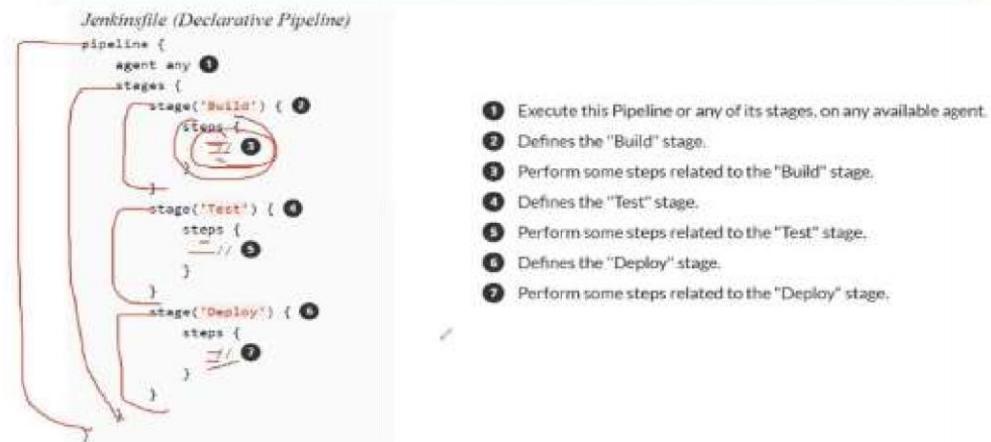
```
node{ stage('Build') { echo 'Want to Build
    project'
}
stage('Test') { echo 'Test the
    Project'
}
stage('Deploy') { echo 'deploy the
    project'
}
}
```

4. Click on Build pipeline

```
node{ stage('Build') { echo 'Want to Build
    project'
}
stage('Test') { echo 'Test the
    Project'
}
stage('Deploy') { echo 'deploy the
    project'
}
}
```



Declarative Pipeline



Dashboard > All >

script pending
+ Recent field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project
Build a maven project, Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Matrix configuration project
projects that need a large number of different configurations, such as testing on multiple environments, platform-specific

Ok



Dashboard > script pipeline > Configuration

Configure

Pipeline

Definition

Pipeline slot: Pipeline slot

```
1+ pipeline {  
2+   agent any  
3+   stages {  
4+     stage('Hello') {  
5+       steps {  
6+         echo 'Hello world'  
7+       }  
8+     }  
9+   }  
10 }  
11  
12 }
```

Save Apply

Stage Logs (Hello)

Print Message -- Hello World (self time 85ms)

Hello World

Build Now Disable Project

Configure Delete Pipeline Full Stage View Rename Pipeline Syntax Build History trend Permalinks

Stage View

Average stage times:
Average full run time: ~50

Now 1827 No Change 571ms

Success 1827 571ms

1827 571ms

Conclusion : Hence we study CI/CD using jenkins server build test deploy jobs.

Devops Exp-5

1) What is the use of maven, gradle, Ant?

→ • Maven : Manages projects builds, dependencies and documentation.

Features - Uses XML configuration, centralized repositories.

When to use : Maven is great for larger projects where dependency management in a structured build process is essential.

• Gradle : A flexible and efficient build system that combines the best of Maven & Ant.

Features : More flexible & efficient than Maven. Allows for incremental builds & parallel task execution.

• Ant : A basic build tool for automating tasks like compiling, packaging & testing.

Features : XML based configurations, highly flexible. Does not have built in dependency management.

When to use : Ant is good for smaller projects or those where a very custom build process is needed.

2) What is pipeline in Jenkins server?

→ A pipeline in Jenkins is a set of automated

processes that defines the steps to build, test, and deploy software. It represents your CI/CD workflows as code. Jenkins pipelines help automate tasks by enable continuous delivery of software by allowing developers to define the entire lifecycle of an application in a structured, reproducible way.

Key concepts of Jenkins Pipelines :

- i) Pipeline as code
- ii) Stages and steps
- iii) Types of pipelines
- iv) Pipeline stages
- v) Pipeline features

Benefits

- i) Automation
- ii) Version control
- iii) Customization
- iv) Scalability

Conclusion : By using Jenkins pipeline teams can streamline their CI/CD processes ensuring faster more reliable releases.

23
4/11/2024



EXPERIMENT NO. 6

Name: Aditi Ambasta

Roll: S004

SAP: 60018220113

Brach: AI & DS

Subject: Devops Laboratory (DJS22ADL5014)

Sem: V

Aim: Setup selenium web driver using Eclipse software

Practical:

Selenium WebDriver Installation

Selenium installation is a 3-step process:

Step 1: Install Java SDK

Step 2: Install Eclipse

Step 3: Install Selenium Webdriver Files

Step 4: Configure Eclipse IDE with WebDriver

In this tutorial, we will learn how to install Selenium Webdriver. Below is the detailed process

NOTE: The versions of Java, Eclipse, Selenium will keep updating with time. But the installation steps will remain the same. Please select the latest version and continue the installation steps below

Step 1 – Install Java Software Development Kit (JDK)

Install Java

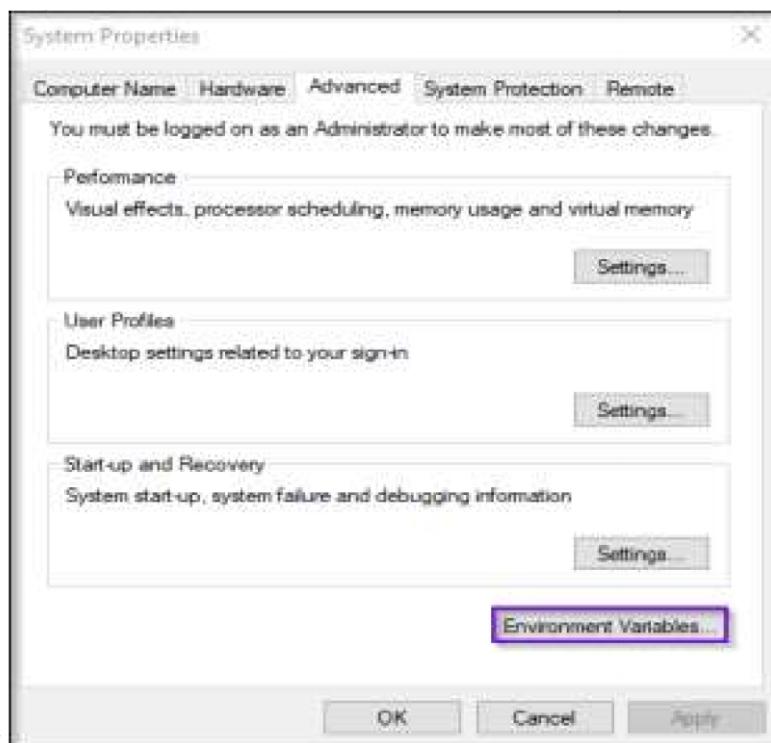
Follow below steps to complete your Java installation.

- Go to the Java Downloads Page and click on the option for Java Platform (JDK).

Product / File Description	File Size	Download
Linux	147.4 MB	jdk-11.0.1_linux-x64_bin.deb
Linux	154.09 MB	jdk-11.0.1_linux-x64_bin.rpm
Linux	171.43 MB	jdk-11.0.1_linux-x64_bin.tar.gz
macOS	166.2 MB	jdk-11.0.1_osx-x64_bin.dmg
macOS	166.55 MB	jdk-11.0.1_osx-x64_bin.tar.gz
Solaris SPARC	186.8 MB	jdk-11.0.1_solaris-sparcv9_bin.tar.gz
Windows	150.98 MB	jdk-11.0.1_windows-x64_bin.exe
Windows	170.99 MB	jdk-11.0.1_windows-x64_bin.zip

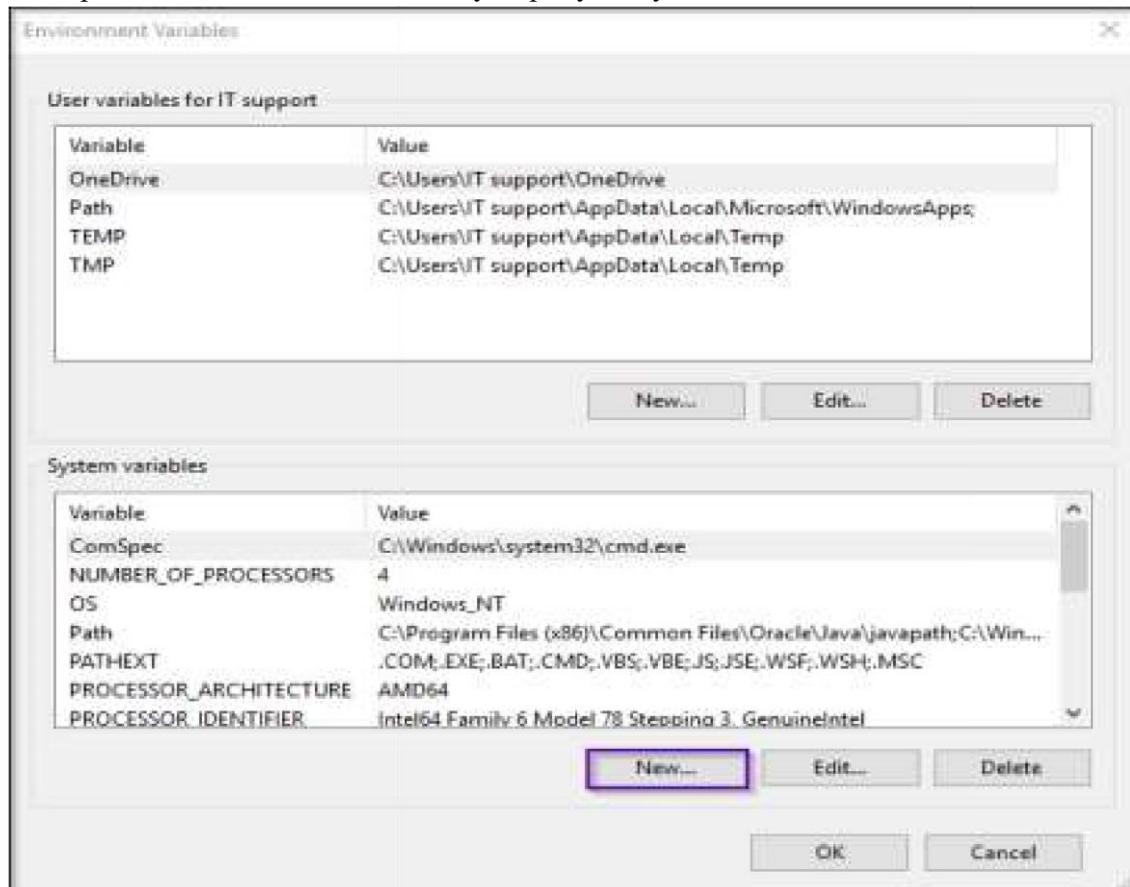
In the next page, select the Accept License Agreement radio button, accept it and click the download link against your matching system configuration.

- You can run the installer once the download is over and follow onscreen instructions.
 - Go to start and search for 'System'
 - Click on 'System'
 - Click on 'Advanced system settings'
 - Click on 'Environment Variables' under 'Advanced' tab as shown

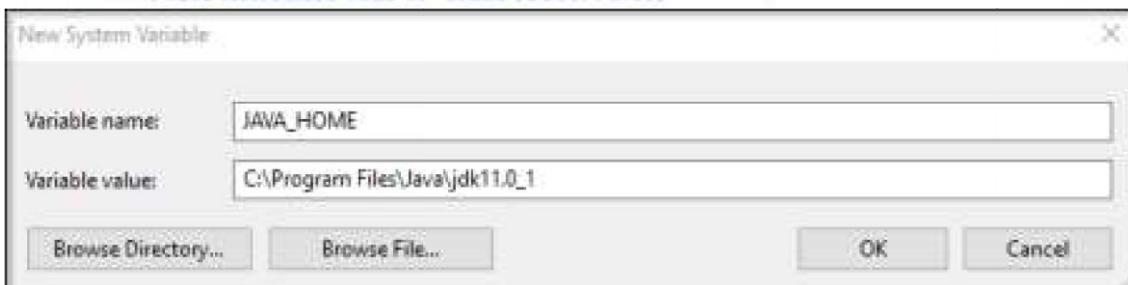


o below:

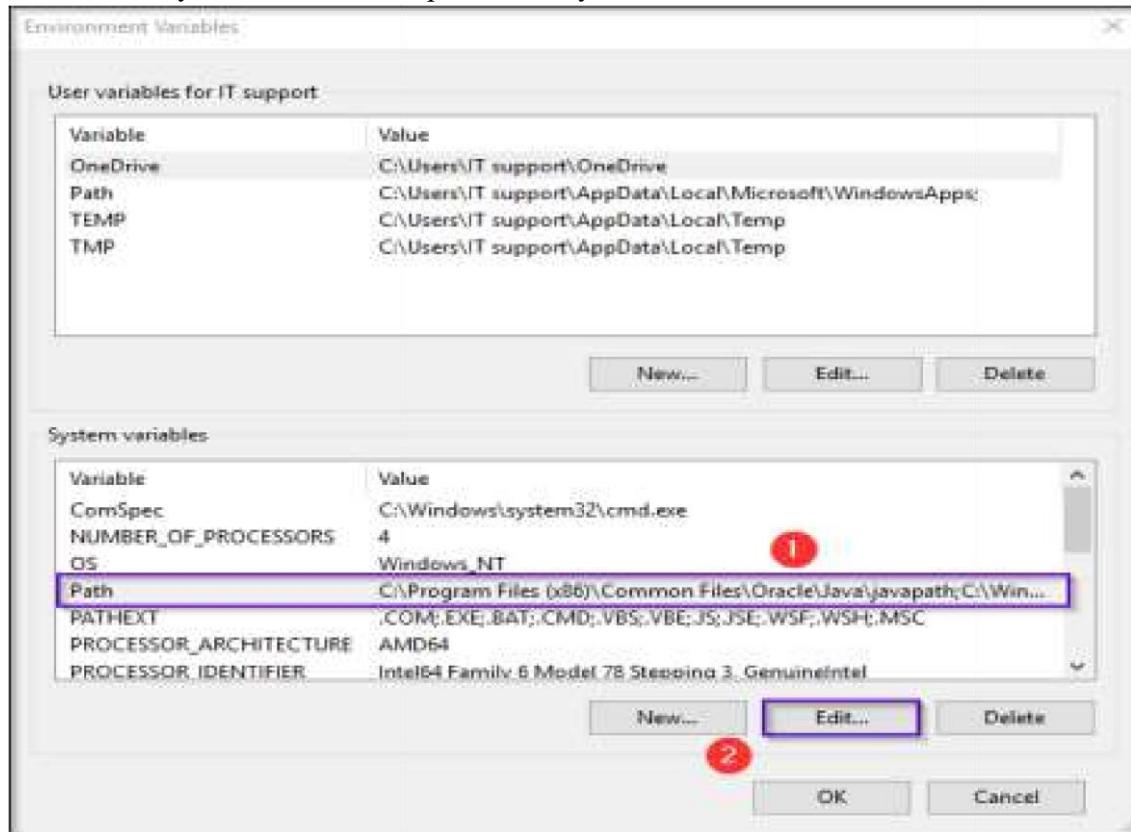
- Next, under system variables choose new and enter the variable name as 'JAVA_HOME' and the full path to Java installation directory as per your system as shown below:



- Below figure depicts the configuration of environment variable name and value.



- Next thing that you have to do is to configure your environment variables. Let's see how to do that. Here, you have to edit the path of the system variable as shown below.



- Under 'Variable value', at the end of the line, enter the following path –
%JAVA_HOME%bin;
Now, you can click 'OK' and you are done.
- Now to cross-check the installation, just run following command in cmd – java -version. It should display the installed version of Java in your system.

Step 2 – Install Eclipse IDE

Download the latest version of “Eclipse IDE for Java Developers” here. Be sure to choose correctly between Windows 32 Bit and 64 Bit versions.



The Eclipse Installer 2022-12 R now includes a JRE for macOS, Windows and Linux.

OpenJDK Runtimes

Get Eclipse IDE 2022-12

Install your favorite desktop IDE packages.

Download Now

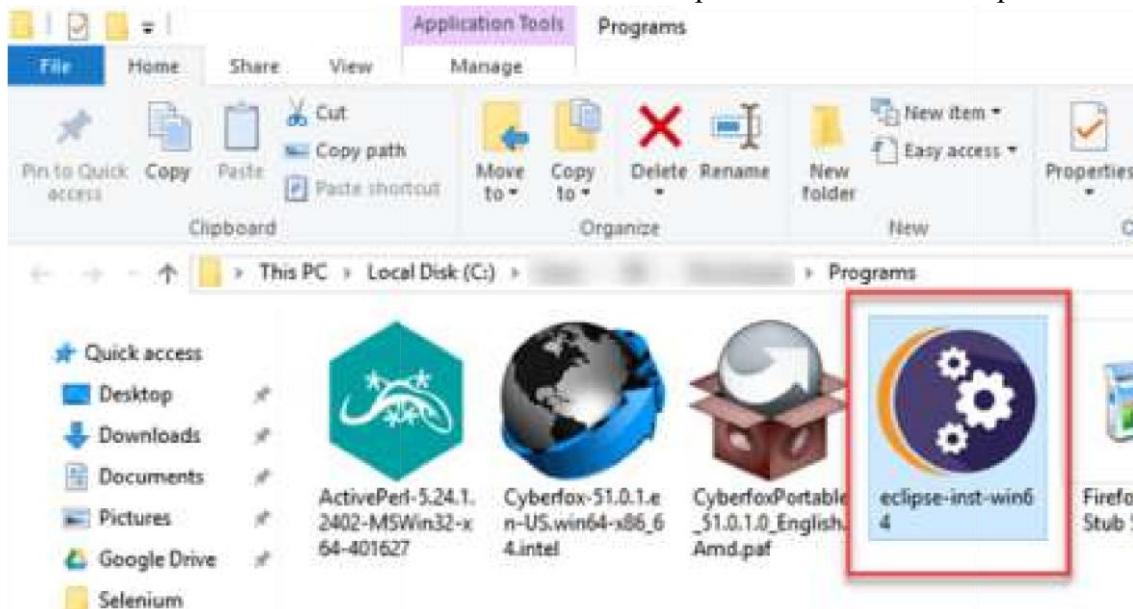
Temurin

The Eclipse Temurin™ project provides high-quality, TCK certified OpenJDK runtimes and associated technology for use across the Java™ ecosystem.

Learn More



You should be able to download an exe file named "eclipse-inst-win64" for Setup.



Double-click on a file to Install the Eclipse. A new window will open. Click Eclipse IDE for Java Developers.



After that, a new window will open which click button marked 1 and change path to "C:\eclipse". Post that Click on the Install button marked 2



After successful completion of the installation procedure, a window will appear. On that window click on Launch.



This will start the eclipse neon IDE for you.

Step 3 – Selenium WebDriver Installation

You can download Selenium Webdriver for Java Client Driver here. You will find client drivers for other languages there, but only choose the one for Java.



This download comes as a ZIP file named "selenium-3.14.0.zip". For simplicity of Selenium installation on Windows 10, extract the contents of this ZIP file on your C drive so that you would have the directory "C:\selenium-3.14.0\". This directory contains all the JAR files that we would later import on Eclipse for Selenium setup.

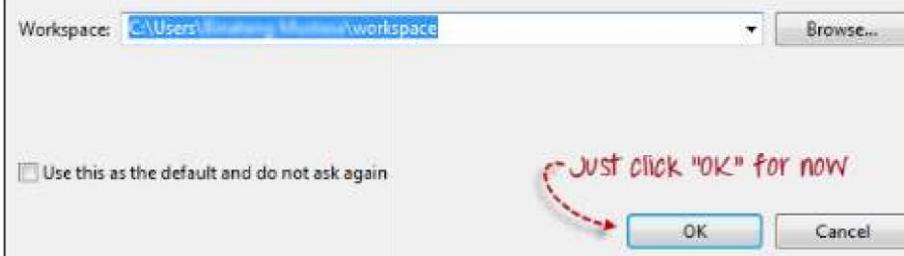
Step 4 – Configure Eclipse IDE with WebDriver

1. Launch the "eclipse.exe" file inside the "eclipse" folder that we extracted in step 2. If you followed step 2 correctly, the executable should be located on C:\eclipse\eclipse.exe.
2. When asked to select a workspace, just accept the default location.

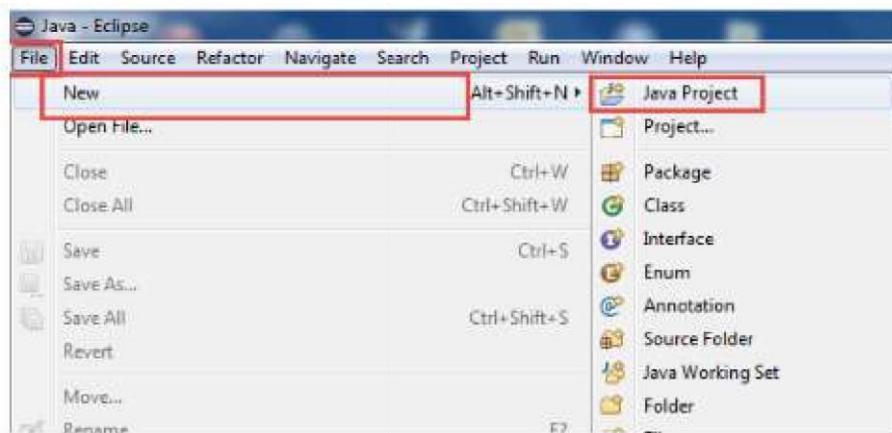


Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

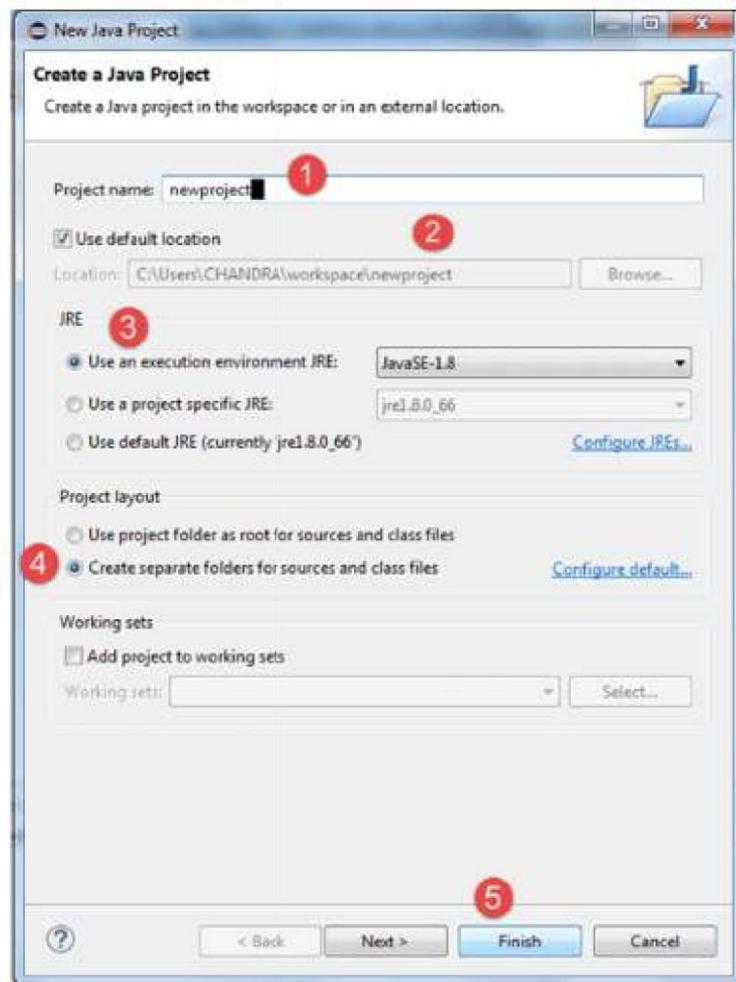


3. Create a new project through File > New > Java Project. Name the project as “newproject”.



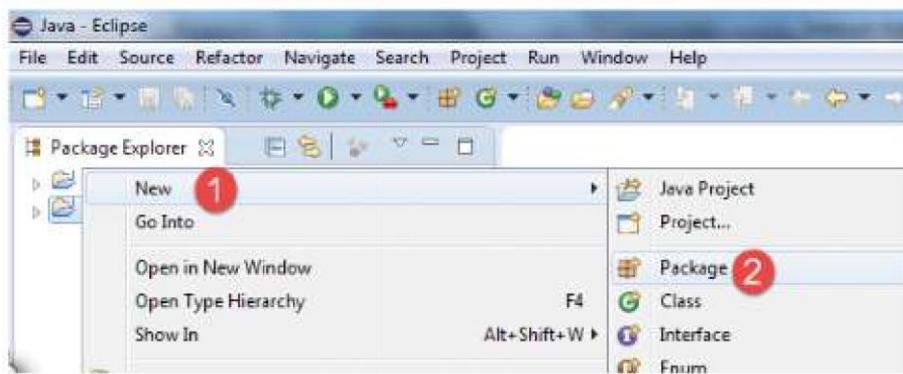
A new pop-up window will open. Enter details as follow

1. Project Name
2. Location to save a project
3. Select an execution JRE
4. Select the layout project option
5. Click on the Finish button



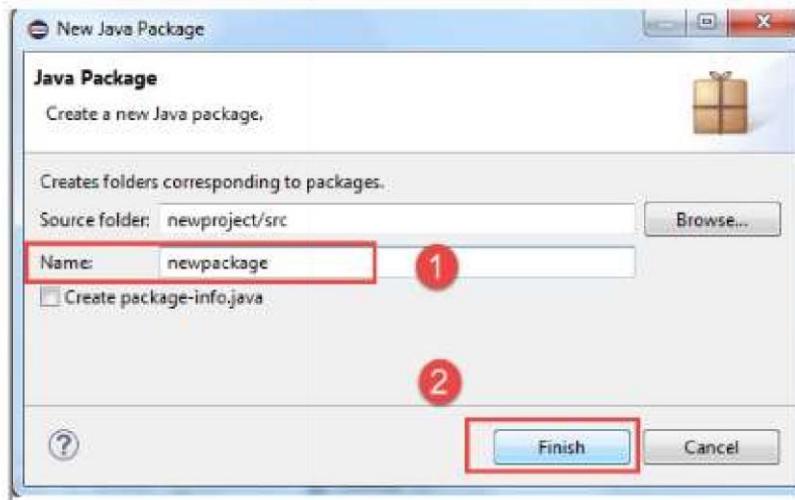
4. In this step,

1. Right-click on the newly created project and
2. Select New > Package, and name that package as “newpackage”.



A pop-up window will open to name the package,

1. Enter the name of the package
2. Click on the Finish button

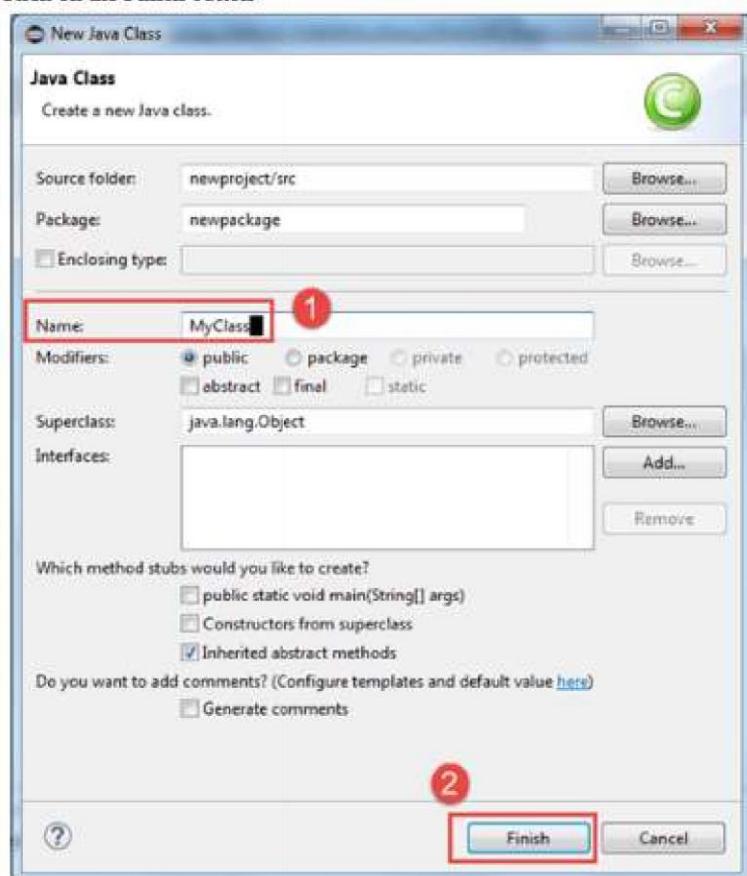


5. Create a new Java class under newpackage by right-clicking on it and then selecting New > Class, and then name it as "MyClass". Your Eclipse IDE should look like the image below.

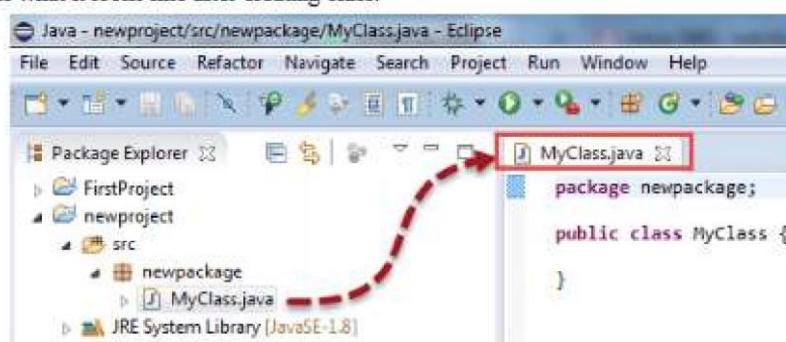


When you click on Class, a pop-up window will open, enter details as

1. Name of the class
2. Click on the Finish button



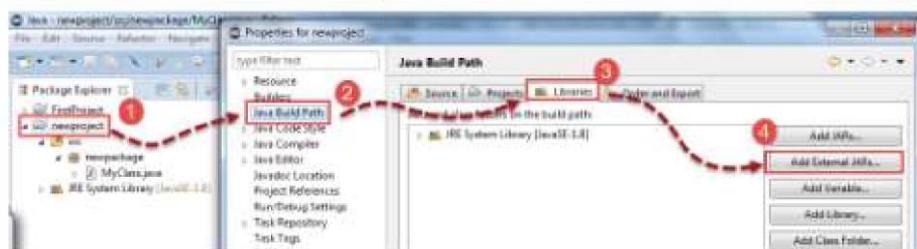
This is what it looks like after creating class.



Now selenium WebDriver's into Java Build Path

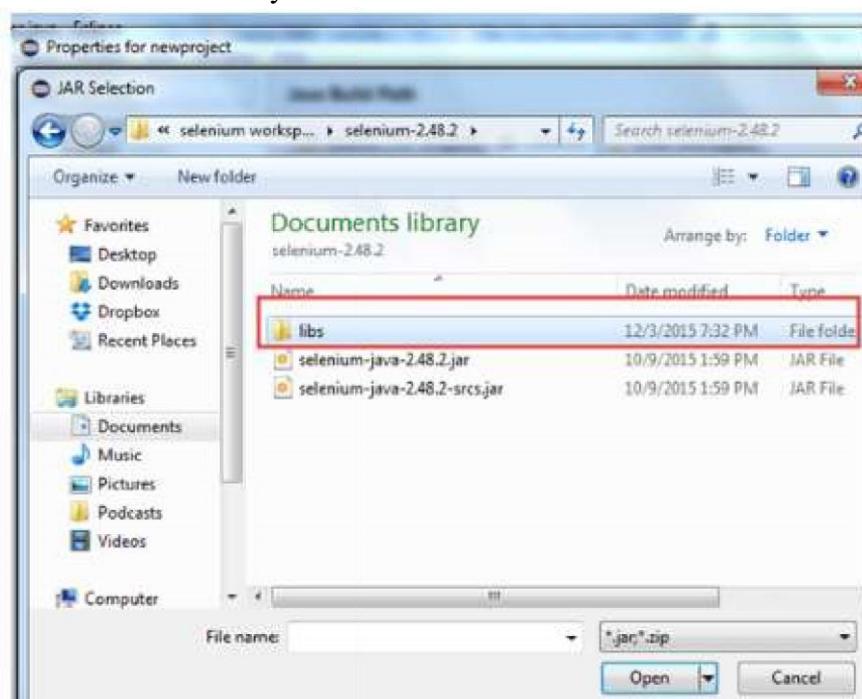
In this step,

1. Right-click on "newproject" and select Properties.
2. On the Properties dialog, click on "Java Build Path".
3. Click on the Libraries tab, and then
4. Click on "Add External JARs.."

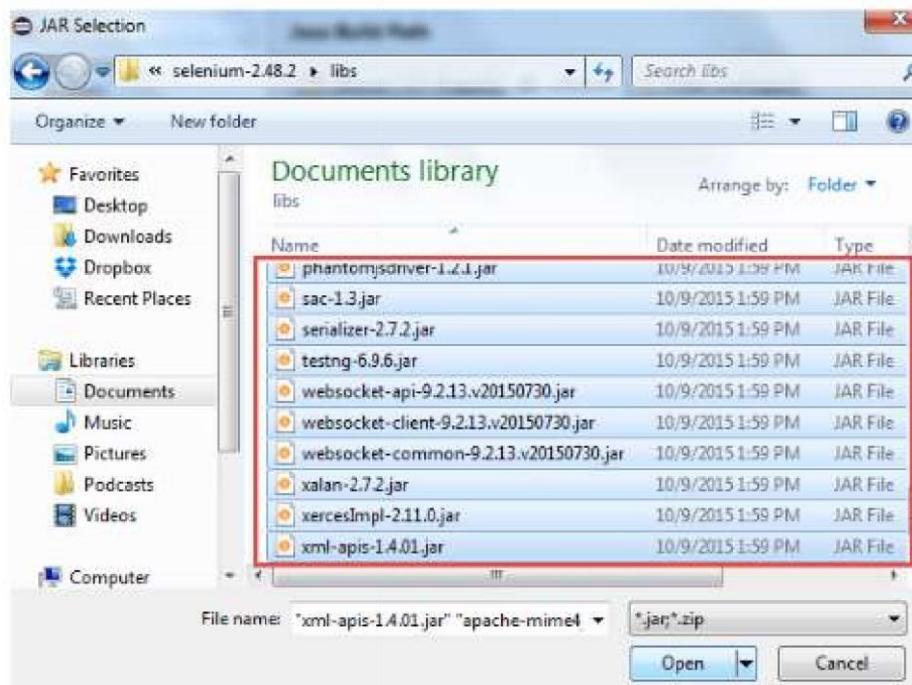




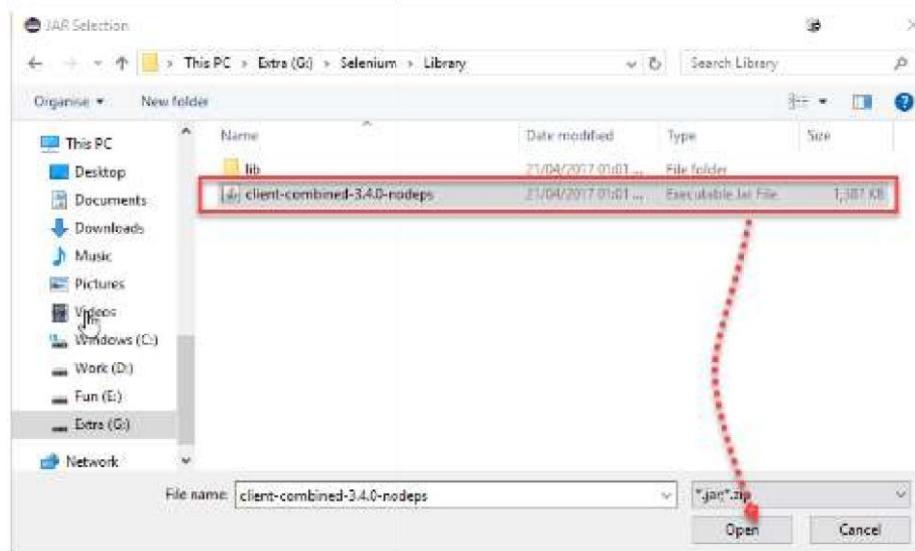
When you click on "Add External JARs.." It will open a pop-up window. Select the JAR files from the selenium folder you want to add.



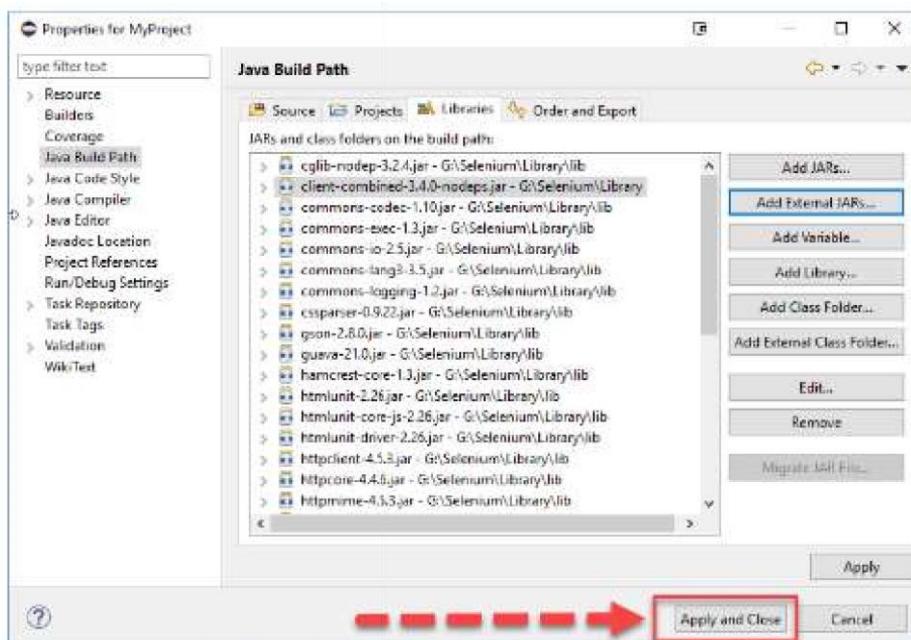
After selecting jar files, click on the OK button. Select all files inside the lib folder.



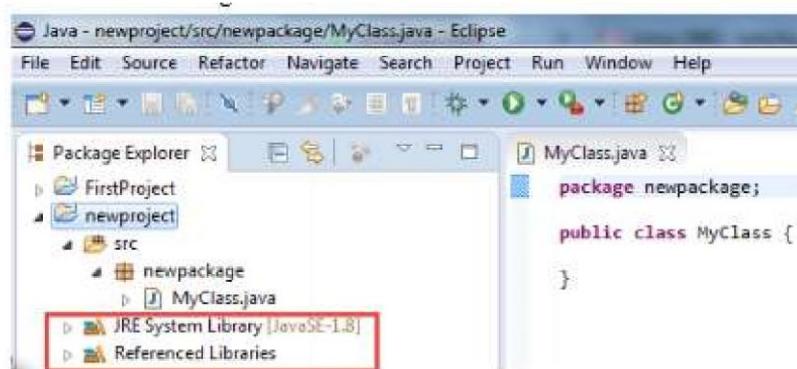
Select files outside lib folder



Once done, click “Apply and Close” button



6. Add all the JAR files inside and outside the “libs” folder. Your Properties dialog should now look similar to the image below.

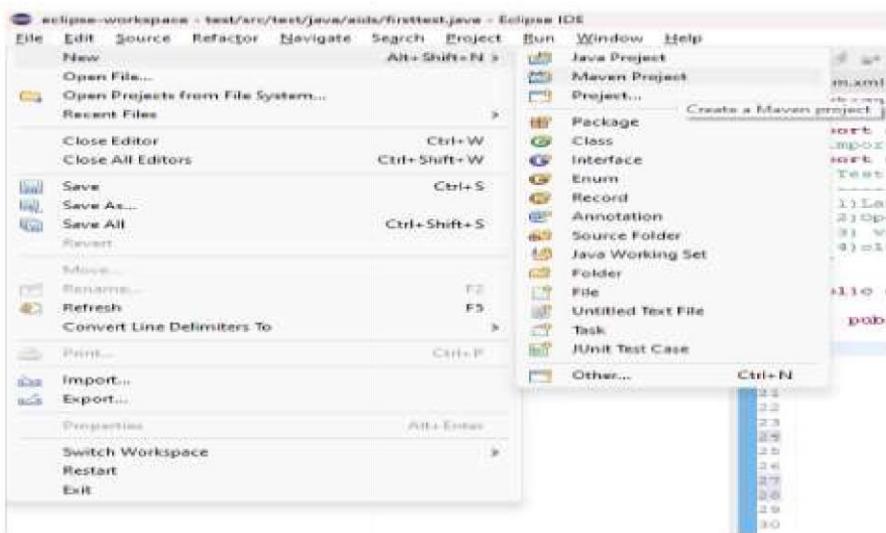


7. Finally, click OK and we are done importing Selenium libraries into our project.

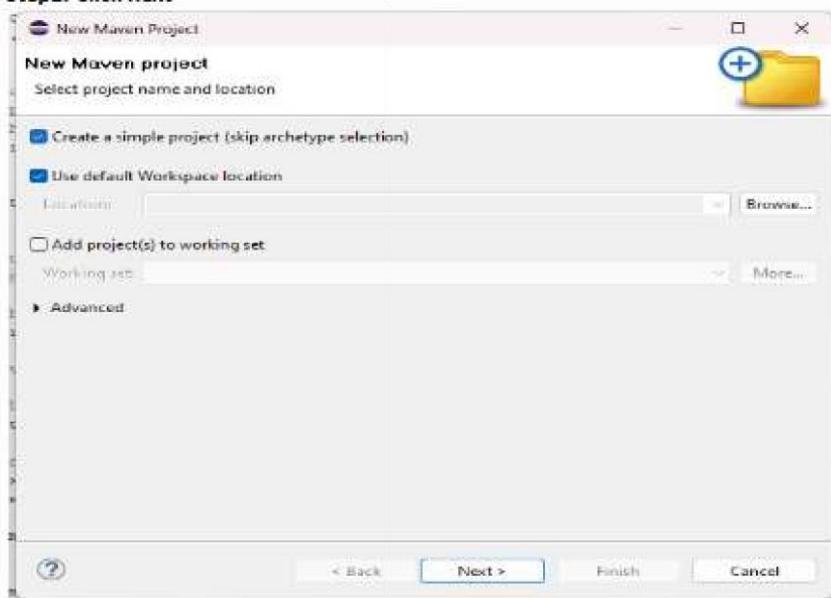


2. Creating Maven Project

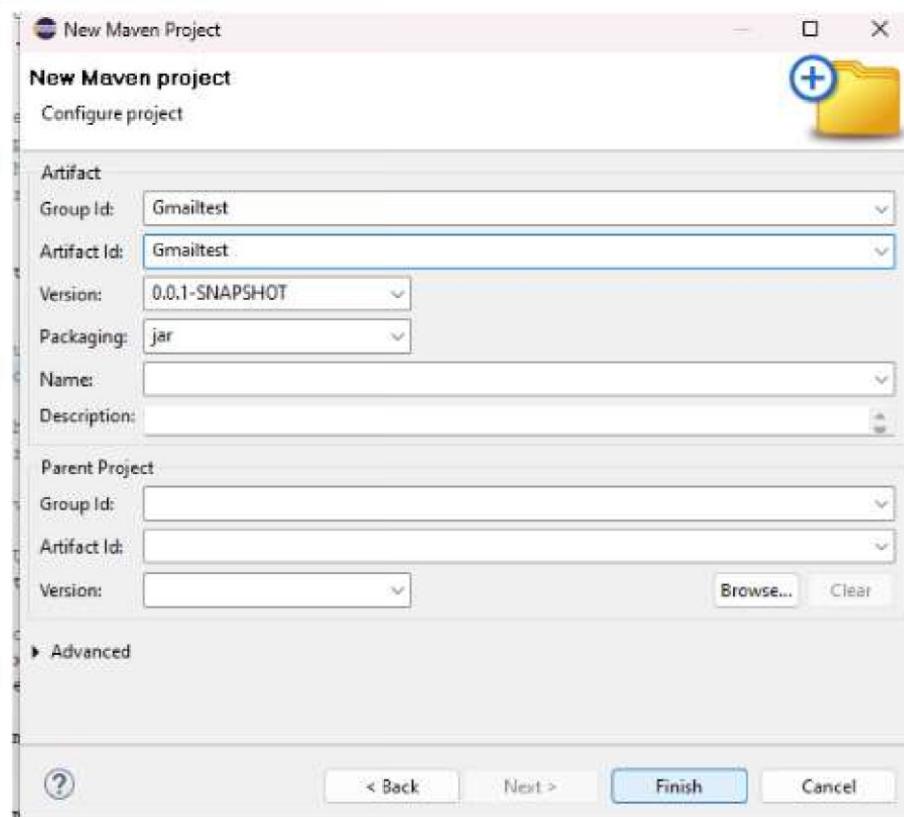
Step1: Create new project



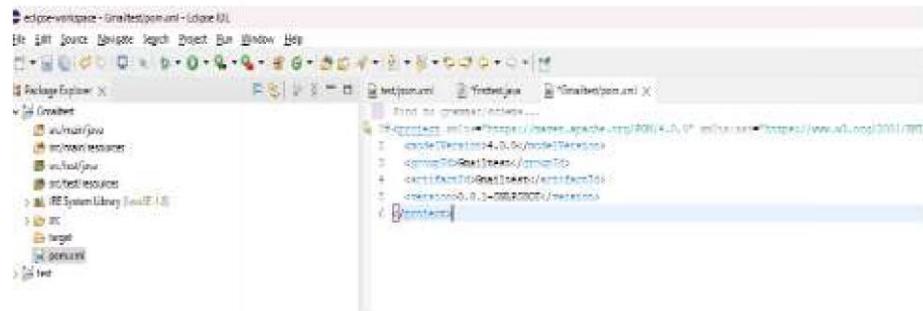
Step2: Click next



Step3: Give name of Group id and click finish



Step4: Open pom.xml and in first line replace http to https



Step5: Configure with Selenium JAR file with eclipse Latest version. Search the "Selenium Java" in the text box.



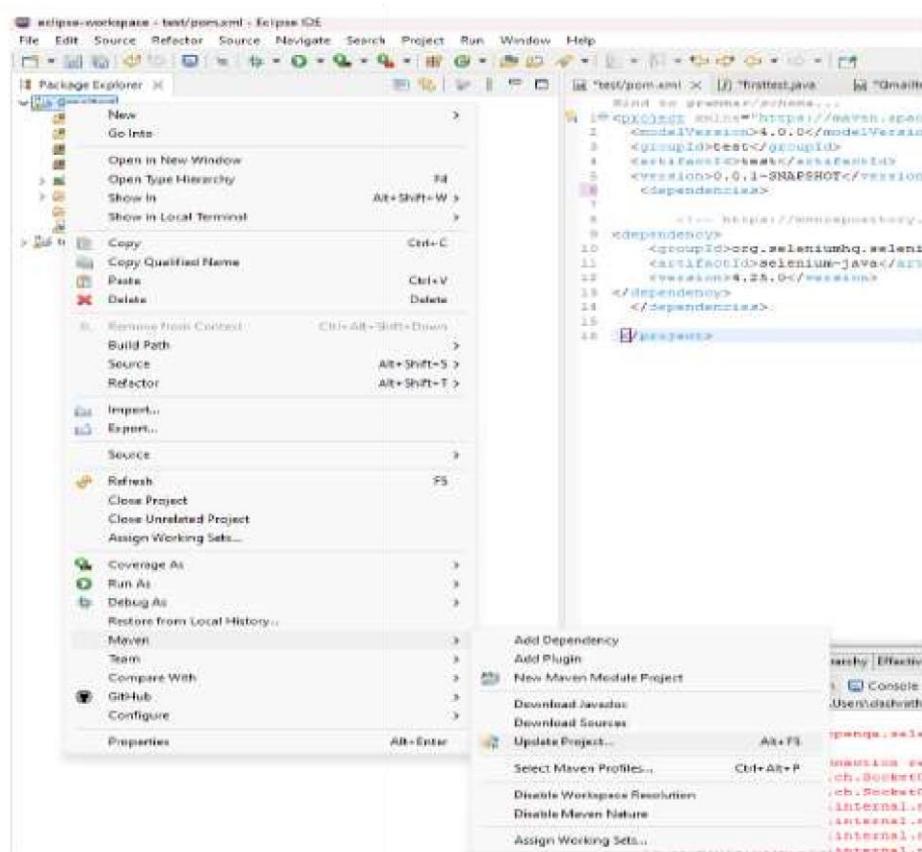
Step6: Copy the dependency code and paste pom.xml

<dependencies>

```
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>4.25.0</version>
</dependency>

</dependencies>
```

Step7: save project next → Right clk on project –maven→ update project



Step8:Right

Clk on src/test/java → create package

Step9: Right Clk on package → create class website

Conclusion: hence we study setup selenium web driver with eclipse editor

Devops, Exp-6

Aim : Setup Selenium webdriven using eclipse software

Theory :

1) Features of Selenium

i) Language Support :

Supports multiple programming language such as Java, Python

ii) Multi Browser capability :

Supports browsers such as chrome, firefox allowing easy testing

iii) Flexible : can be executed in linux, MAC etc.

iv) Community Support : Large Documentation & resource available

v) Open source : freely available for everyone to use

vi) Performance & Speed : Web drivers automate web testing allowing to execute test case quickly.

2) Explain different software testing automation tools

→ i) Browser stack Automate : Web testing product which helps run application tests on all desktop browsers. It includes automate scale by test management, accessibility testing to ensure that you have an end to end solution for all software

ii) Selenium : Developed by Jason Higgins in 2004, which is open source web user interface used for testing software. Software is developed on Linux, Mac by windows.

iii) Bug Bug : Web testing tool for test automation unlike conventional testing tools. technical by non technical users. It offers an intuitive interface for creating, editing, and executing list without coding.

Conclusion : Hence, here we have successfully learnt about the different devops tools used during testing by successfully setup selenium webserver using eclipse edition.

29

5/11/2021



EXPERIMENT NO. 7

Name: Aditi Ambasta

Roll: S004

SAP: 60018220113

Brach: AI & DS

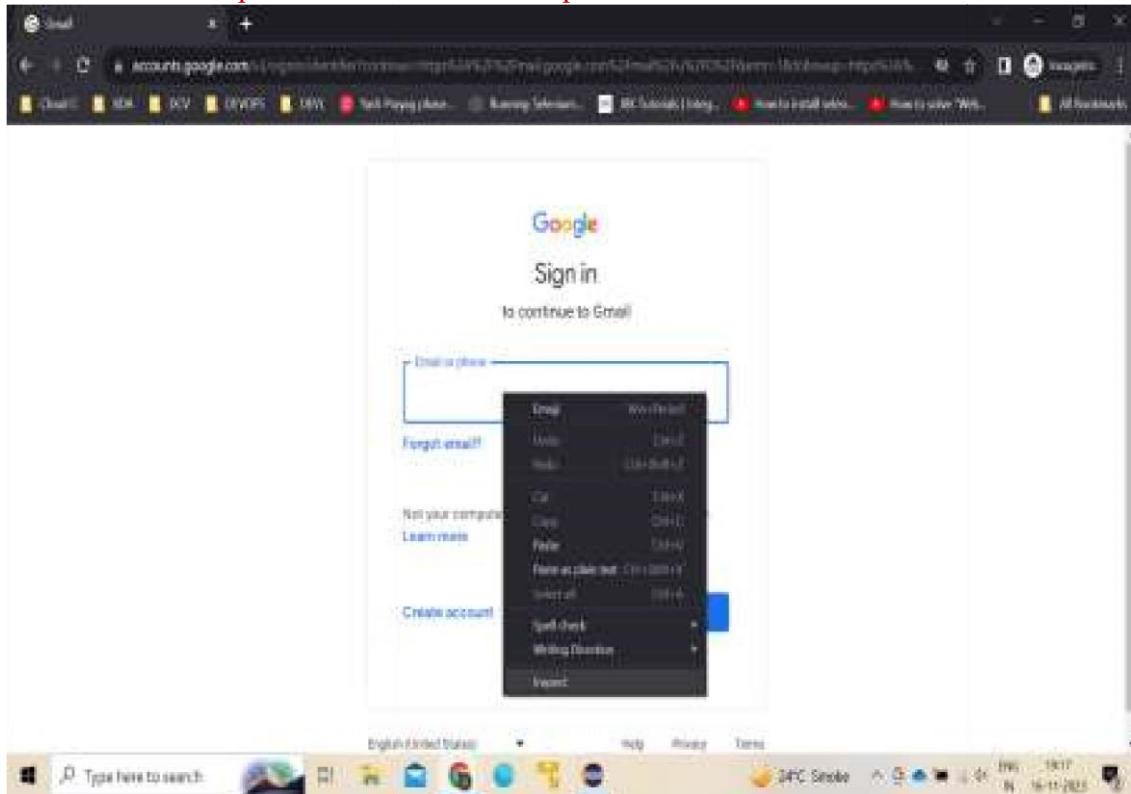
Subject: Devops Laboratory (DJS22ADL5014)

Sem: V

Aim: Run Gmail Signing Page test script using Selenium.

Practical:

Two methods to implement selenium test Script.



1. Downloading JAR files and attaching them to java projects (manually) – Not recommended.

Step1. Install Latest webdriver available in given [link](#)

Step2: Provide WebElement value(either id or name) for Email or phone textbox



The screenshot shows a web browser window with the Google Sign-in page loaded. The URL in the address bar is <https://accounts.google.com/ServiceLogin?hl=en&continue=https://www.google.com/>. The page features the Google logo and a 'Sign in' button. A yellow box highlights the 'Next' button. A context menu is open over the 'Next' button, with the 'Copy element' option selected. The menu also includes 'Copy value', 'Copy href', 'Copy to clipboard', 'Copy style', 'Copy title', and 'Copy full value'. The browser's status bar at the bottom shows the date as 16-11-2023.

Step:3 Click on Next Button and fidget element value



A screenshot of a Microsoft Edge browser window. The address bar shows a Google sign-in URL. The main content is a Google 'Sign in' page for Gmail. A context menu is open in the center of the page, listing standard browser actions like Back, Forward, Stop, Reload, and various tab management options. At the bottom right of the menu, the 'Next' button on the sign-in form is highlighted with a red rectangular box. The browser interface includes a title bar with the Microsoft Edge logo and a standard window control bar (minimize, maximize, close).



Step4: Write given code in Eclipse editor

```
package aids;
import java.util.concurrent.TimeUnit; import
org.openqa.selenium.By; import
org.openqa.selenium.WebDriver; import
org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.*;
public class gmail1 { public static void main(String[] args) {
    // TODO Auto-generated method stub
    //System.setProperty("webdriver.chrome.driver","C:\\Web
Driver\\chromedriver-win64\\chromedriver.exe"); WebDriver driver= new
    ChromeDriver(); driver.get("https://www.gmail.com");
    System.out.println(driver.getTitle());

    driver.findElement(By.id("identifierId")).sendKeys("softesting001@gmail.com
");
    driver.findElement(By.id("identifierNext")).click();
    driver.findElement(By.id("password")).sendKeys("aids@123");
    driver.findElement(By.id("passwordNext")).click(); driver.quit(); driver.close();
}
}
```

Step5: Click on run Button

2.Creating Maven Project

Reference of Experiment no6

Step9: Right Clk on package→ create class website(firsttest)

```
package aids;
import org.openqa.selenium.chrome.ChromeDriver;
//import org.openqa.selenium.edge.EdgeDriver; import
org.openqa.selenium.WebDriver;
/* Test Case
* -----
* 1)Launch Browser(chrome)
* 2)Open URL https://www.msbte.com
* 3) validate the title should be "Home |
§" * 4)close browser
*/
public class firsttest { public static void main(String[] args) {
    // TODO Auto-generated method stub
    //System.out.println("dashrath kale");

    //1)Launch Browser(chrome)
    //ChromeDriver driver=new ChromeDriver(); WebDriver driver=new
    ChromeDriver();
    // WebDriver driver=new EdgeDriver();
```



```
//2)Open URL https://www.msbte.com
driver.get("https://www.iitb.ac.in/");

//3) validate the title should be "Home |
\x" String text=driver.getTitle(); if
(text.equals("Home |
\x"))
{
    System.out.println("Test Case Pass");
} else {
    System.out.println("Test Case Fail");
    } driver.close();
//driver.quit();

}
}
```

Conclusion: Hence we study how run test script with help of Selenium web driver

DevOps - Exp 7

1. What types of testing are done in CD on the test server in production & explain any one

→ On a Continuous Deployment (CD) pipeline, testing on a test server in a production-like environment is critical for ensuring that the application is ready for release without manual intervention. The key types of testing done in CD on test server are:

- ① Smoke testing
- ② Sanity testing
- ③ End-to-End testing
- ④ User acceptance testing
- ⑤ Performance testing

User Acceptance Testing (UAT) is a type of testing performed in the final phase of the CD pipeline, typically on test server that mimics the production environment. The primary purpose of UAT is to validate whether the software meets the business requirements & is ready for release to production.

~~Detailed explanation of UAT.~~

- ① End-to-end user focus
- ② Business requirement validation
- ③ Real world scenarios
- ④ Approval for release

Advantages,

- ① Ensures that the product meets the expectations of users.

② Reduces the risk of post-release issues as actual business scenarios are tested.

2. What types of testing are done in CI build?

→ In CI build, various types of testing are done to ensure the code is functional, reliable & maintainable. Key types of testing are,

- ① Unit testing
- ② Integration testing
- ③ Acceptance testing
- ④ Performance testing
- ⑤ Smoke testing

Unit testing is the process of testing individual units or components of a software application to ensure that each part of the code functions as expected. Frameworks & tools are JUnit and pytest. Advantages are early bug detection & documentation.

Conclusion: Thus CI & CD play important role for testing.

Act 23
5/11/2024

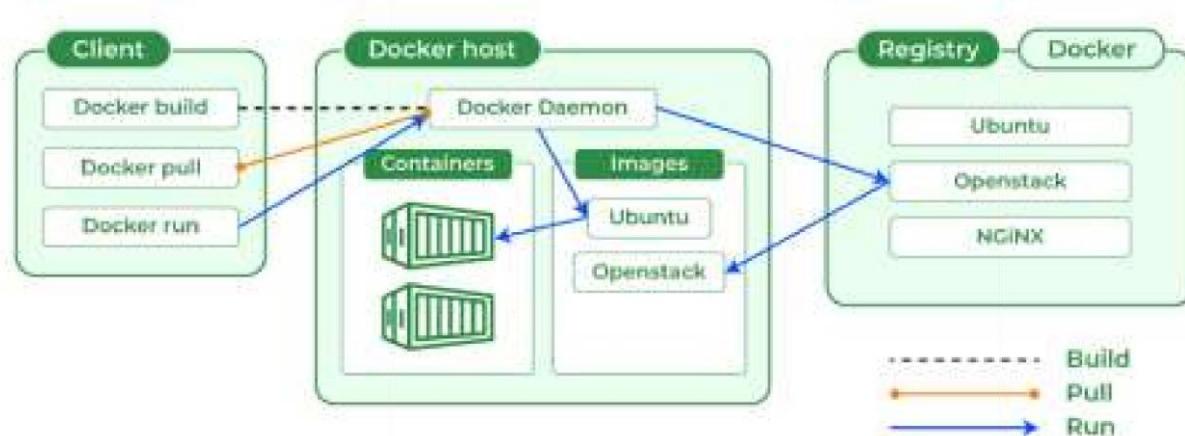


EXPERIMENT NO. 8

Name: Aditi Ambasta Roll: S004 SAP: 60018220113 Brach: AI-DS Subject: Devops Laboratory (DJS22ADL5014) Sem: V

AIM: Create Docker Image for to implement ETL pipeline with mysql workbench using python.

Practical:



Install Docker Desktop

Step 1: Download Docker Desktop

1. Go to the official Docker website: Docker Desktop for Windows.
2. Click Download Docker Desktop for Windows.

Step 2: Install Docker Desktop

1. Locate the downloaded .exe file in your Downloads folder and double-click to start the installation.
2. Follow the installer's instructions:
 - Click OK to continue.
 - The installer will download and install the necessary components.
3. After the installation completes, restart your system.

Step 3: Launch Docker Desktop

1. After restarting, find Docker Desktop in your Start Menu and launch it.
2. During the first run, Docker Desktop may ask for administrative privileges to configure components. Grant the permission.
3. Docker Desktop should now launch and show a window indicating that it's running.



Step 4: Verify Docker Installation

Open Command Prompt or PowerShell and run the following command to verify Docker is running:

```
docker --version
```

You should see something like:

```
Docker version 24.0.0, build abcdef1 Test
```

Docker by running a simple container:

```
docker run hello-world
```

1. If Docker is installed correctly, you should see a message saying that the Docker installation is working.

```
C:\code>docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:d211f485f2dd1dee407a80973c8f129f00d54604d2c90732e8e320e5038a0348
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.
```

Step 5:Prerequisites

1. Install Python 3.12: Ensure that Python 3.12 is installed on your Windows machine. You can download it from [the official Python website](#). Remember to check the option to add Python to your PATH during installation.
2. Install MySQL Server: Download and install MySQL Server from [MySQL's official site](#). During installation, set up your root password and remember it.
3. Install MySQL Workbench: Download and install MySQL Workbench from [here](#).
4. Install Required Python Libraries: You will need [mysql-connector-python](#) for connecting to MySQL and possibly [pandas](#) for data transformation. You can install them using pip:

Step 6: Create database

```
CREATE DATABASE exp10;
USE exp10;
```



```
CREATE TABLE transformed_data ( id INT  
AUTO_INCREMENT PRIMARY KEY, name  
VARCHAR(100),  
value INT,  
transformed_value INT  
);  
SELECT * FROM transformed_data;
```

Step 7:

pip install mysql-connector-python pandas pip install pyarrow #will improve the performance of certain Pandas operations. Step 8: Set Up

Your MySQL Database

1. Open MySQL Workbench.
2. Connect to the MySQL Server using the root user or any other user you've created.
3. Create a New Database:
 - In the Navigator panel, right-click on Schemas and select Create Schema.
 - Name your schema (e.g., [etl_pipeline](#)) and click Apply.
4. Create Tables: You need to create tables for storing your extracted and transformed data. You can execute SQL commands in the SQL Editor.

Step 9: Extract Data from Source

You can extract data from various sources like CSV files, APIs, or other databases. For this example, we'll extract data from a CSV file.

Create a Sample CSV File: Create a CSV file (e.g., [1.csv](#)) with the following content:

```
name,value  
Ram,100  
Sham,150  
Radha,200  
Durga,150
```

Step 10: give file name ETL.py

```
import mysql.connector import  
pandas as pd
```

```
# Extract
```

```
#Windows path df = pd.read_csv(r'C:\Users\admin\Downloads\1.csv') df =  
pd.read_csv('/project/data/1.csv') # docker unable to access local file so that option to  
#mount your local directory into the container.
```

```
# Transform
```



```
df['transformed_value'] = df['value'] * 2

# Load
connection = mysql.connector.connect(
# host='localhost',
host='host.docker.internal', # Docker not support direct local host
user='root', # Replace with your MySQL username password='das@11',
# Replace with your MySQL password database='exp10'
)

cursor = connection.cursor()

for index, row in df.iterrows():
    cursor.execute("INSERT INTO transformed_data (name, value, transformed_value) VALUES (%s, %s, %s)", (row['name'], row['value'], row['transformed_value'])) query
    = "SELECT * FROM transformed_data"

# Step 3: Use pandas to execute the query and store the result in a DataFrame df =
pd.read_sql(query, connection)

# Step 4: Print the DataFrame or work with the data print(df)

connection.commit() # Commit the transaction
cursor.close() connection.close()
```

Step 11: Create dockerfile with .txt extension in notepad

```
# Use the official Python Alpine image FROM
python:alpine

# Set the working directory in the container
WORKDIR /project

# Copy the current directory contents into the container at /project
COPY . /project

# Install dependencies if you have a requirements.txt file RUN
pip install -r requirements.txt

# Run the Python script when the container launches CMD
["python", "1.py"]
```



Step 12: Create requirements.txt file in notepad Mentioned

below given python library files

mysql-connector-python pandas

Step 13: Go on CMD prompt Run command

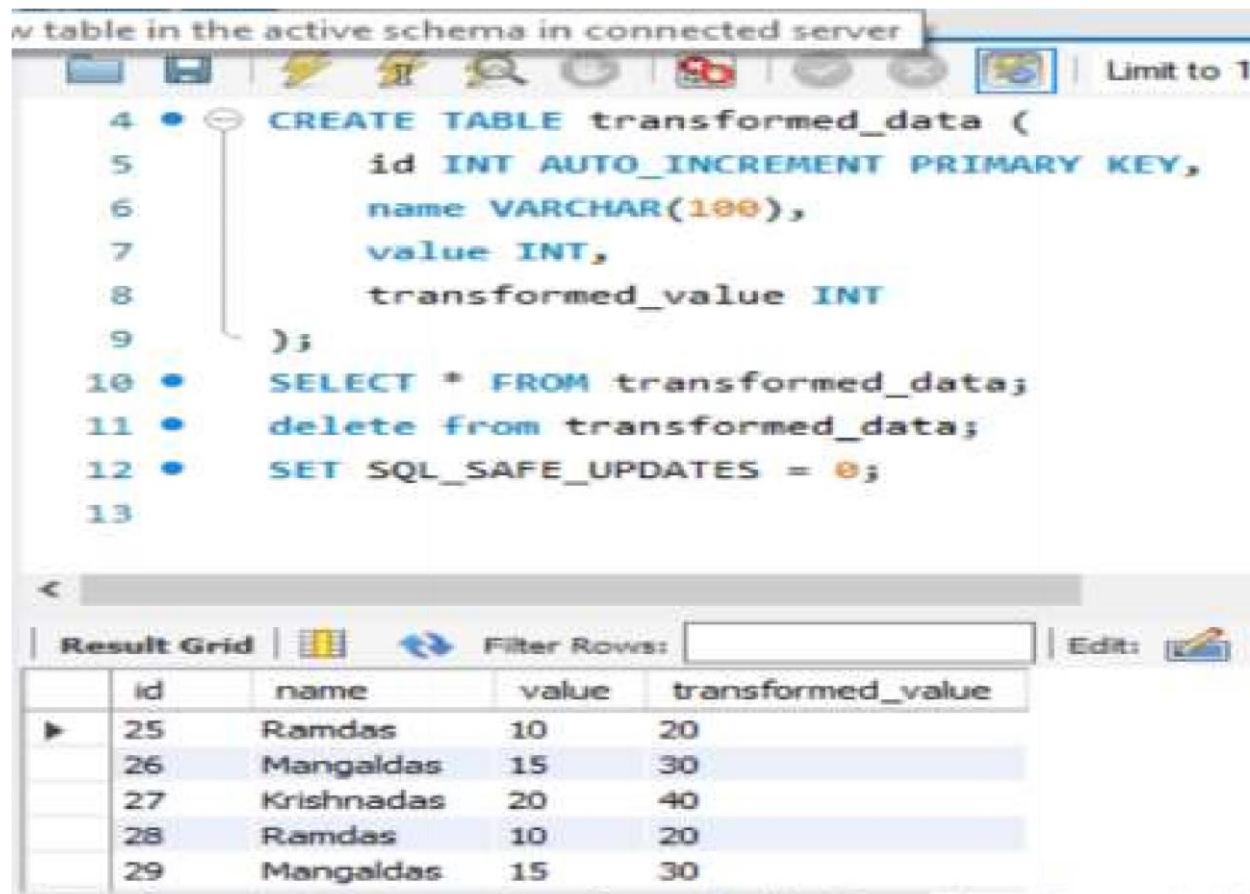
C:\code>docker build -t djimage -f dockerfile . #docker build the image name djimage

C:\code>docker run djimage # it will throws error bcz 1.csv yet to mount in container C:\code>docker run -v C:/Users/admin/Downloads:/project/data djimage

Step 14: Check data extract from 1.csv and transformed, loaded to SQL server



~ table in the active schema in connected server



```
4 • CREATE TABLE transformed_data (
5     id INT AUTO_INCREMENT PRIMARY KEY,
6     name VARCHAR(100),
7     value INT,
8     transformed_value INT
9 );
10 •     SELECT * FROM transformed_data;
11 •     delete from transformed_data;
12 •     SET SQL_SAFE_UPDATES = 0;
13
```

Result Grid

	id	name	value	transformed_value
▶	25	Ramdas	10	20
	26	Mangaldas	15	30
	27	Krishnadas	20	40
	28	Ramdas	10	20
	29	Mangaldas	15	30

```
C:\code>docker run -v C:/Users/admin/Downloads:/project/data djimage
/project/1.py:32: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection)
) or database string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
df = pd.read_sql(query, connection)
   id      name  value  transformed_value
0  25    Ramdas     10            20
1  26  Mangaldas     15            30
2  27  Krishnadas     20            40
```

Docker images



```
C:\code>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
djimage	latest	c7db28a68a7c	2 hours ago	234MB
<none>	<none>	979e5f00ae35	2 hours ago	234MB
<none>	<none>	7b68f5958186	2 hours ago	234MB
<none>	<none>	9f13b766c472	2 hours ago	56.1MB
<none>	<none>	3ff9de44f365	2 hours ago	44.6MB
hello-world	latest	d2c94e258dcb	17 months ago	13.3kB

```
$sudo service docker stop
```

```
$sudo service docker start
```

Conclusion: Hence we study how to install docker and deploy projects in any environment.

Devops - Exp 8

1. Describe the architecture of Docker & explain every component in detail.

→ Docker architecture is a client-server architecture consisting of three main components. It is built to simplify the creation, deployment & management of applications in lightweight containers.

① Docker client : The user-facing interface that sends commands to Docker Daemon (via CLI or Rest API) common commands include docker run, docker pull & docker build

② Docker Daemon (Docker Engine) : the background service that does the heavy lifting like creating, running & managing Docker containers. It listens to requests from the Docker client & handles container orchestration, image management & handles functions.

③ Docker objects :
Images : Immutable templates for creating containers
Volumes : Persistent storage that containers can use to store & share data.

Networks : Allow communication between Docker containers & external systems.

2. Explain Nagios architecture in detail.

→ Nagios architecture is based on a client-server model & includes following key components:

① Nagios core: The central server component responsible for scheduling & processing checks. It uses plugins to gather data from monitored devices. The core processes this data & generates alerts & reports.

② Plugins: Small progs that run on Nagios server to collect info about status of services.

③ Nagios Remote Plugin Executer (NRPE): A plugin that allows Nagios to execute plugins on remote hosts for services that can't be monitored directly.

④ Web interface: A user interface that provides access to system status, reports & logs.

⑤ Notification System: It sends all alerts (via emails, SMS, etc.) when issues are detected.

24

24

4111111111

Aditi Ambasta
60018220113
8004

DevOps Assignment - 01

1.

→ What is configuration management?

Configuration management (CM) tools help automate the process of tracking, management controlling software configuration across different environment. They ensure that software remains functional & convenient when moved from one environment to another.

Popular CM tools

- i) Ansible :- Open source automation tool used for CM and application deployment
- ii) Chef :- Automates infrastructure management through code
- iii) Salt Stack :- CM and Orchestration tool with real time event driven automation
- iv) Terraform :- Infrastructure tool that manages cloud & resources.

2. Iterative Model

- focuses on repeating the development cycles and delivering portion to refine and enhance the product
- Each iteration goes through entire SDLC

Incremental Model

- focuses on developing portion of the product separately.
- Each increment is a stand alone part of final product.

- Feedback is incorporated in each iteration
- Delivery is often done at end of iteration cycle.
- Cycle refinement of the entire system
- feedback only integrated after increment.
- Delivery occurs with each increment & can affect subsequent increments.
- linear development in small increments.

Containers

Lightweight portable encapsulation of app & their dependencies.

Shares the host OS

More efficient, uses less overhead

Significantly much faster

Less startup time

Virtualization

- Technology that allows multiple storage & operating system to run on single machine
- Provides full isolation with its own OS.
- less efficient due to the more overhead due to running full OS instances.
- Slower to execute as the programs are heavier.
- Extremely high startup time due to the OS Boottime.

4.

Merge

- Retains the original commit history, creating a merge commit that reflects conflict resolution.
- Can handle conflicts during merge process & creates a merge commit that reflects conflict resolution.
- Easier for collaborative workflows, especially with multiple developers.
- Branch history remains diverged, showing where branches split & then are re-merged.
- Best for maintaining records of branch merges, preserving the history.
- Rewrite the commit history to create a linear history.
- Conflict can be resolved for each individual commit when rebasing.
- Requires more care especially in shared branch as it rewrite commit history.
- creates a clean, linear commit structure, avoiding the original branch divergence.
- Best for creating a linear history in private branch or solo work.

5.

Diff - Scrum & Extreme Programming (XP)

Scrum

- Agile framework for managing and completing projects.

Extreme Programming

- Mainly focused on best engineering practices.

- Uses sprint - typically 2-4 weeks long.
- Defines specific roles (Product Manager, owner), emphasizes team work, scrum master, team developers
- Emphasizes scrum ceremonies (Daily standups, sprint reviews)
- Minimal documentation focuses mainly on the successful implementation of a working project
- Uses short iterations, often 1-2 weeks.
- Fewer emphasized roles, emphasizes team work.
- Emphasizes technical practices (pair-programming, test-driven optimization)
- Minimal with emphasis, on focus on code and even clear communication.

6.

Fetch

- Downloads changes from a remote repository to local repository without merging them.
- To update local repo with remote changes without affecting dir.
- No direct impact requires manual merge.
- Allows us to review changes before merge.
- eg:- ~~git fetch origin~~

Pull

- Downloads changes from a remote repository and even merges changes with local branch.
- Update local repo while simultaneously integrating changes from remote repo.
- Directly updates and even merge with local branch.
- combinations fetching and merging into single step.
- ~~git pull origin main~~