EXPERIMENT-1

AIM: To write a code for a simple user registration form for an event.

PROGRAM:

HTML CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>User Registration</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="container">
    <h2>User Registration</h2>
    <form action="/submit-registration" method="post">
      <div class="form-field">
        <label for="firstname">First Name:</label>
        <input type="text" id="firstname" name="firstname" required>
      </div>
      <div class="form-field">
        <label for="lastname">Last Name:</label>
        <input type="text" id="lastname" name="lastname" required>
      </div>
      <div class="form-field">
        <label for="mobile">Mobile Number:</label>
        <input type="tel" id="mobile" name="mobile" pattern="[0-9]{10}" required>
      </div>
      <div class="form-field">
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required>
      </div>
      <div class="form-field">
        <label for="password">Password:</label>
        <input type="password" id="password" name="password" required>
      </div>
      <div >
        <input type="checkbox" id="terms" name="terms" required>
        <label for="terms" class="terms-condition">I agree to the Terms &
Conditions</label>
      </div>
```

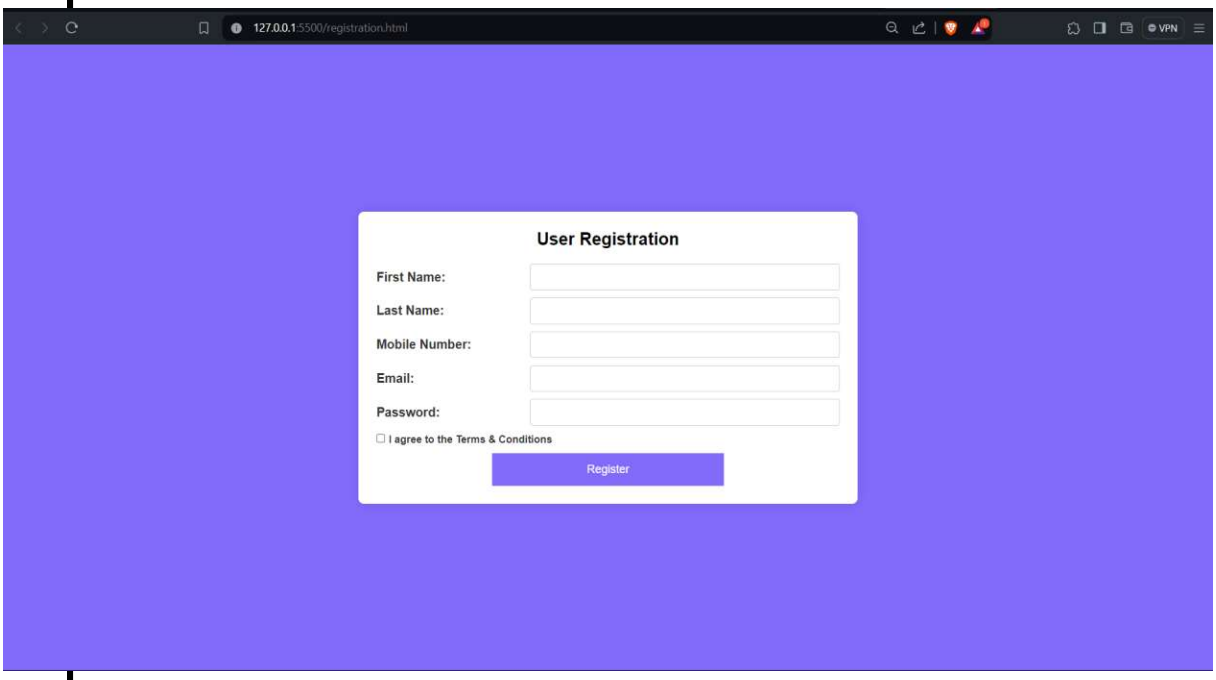
```
        <input type="submit" value="Register">
      </form>
    </div>
</body>
</html>
```

CSS CODE:

```
* {
   margin: 0;
   padding: 0;
   box-sizing: border-box;
   font-family: "Poppins", sans-serif;
}
body {
   min-height: 100vh;
      display: flex;
      align-items: center;
      justify-content: center;
      padding: 20px;
      background: rgb(130, 106, 251);
}
.container {
   position: relative;
   max-width: 700px;
   width: 100%;
   background: #fff;
   padding: 25px;
   border-radius: 8px;
   box-shadow: 0 0 15px rgba(0, 0, 0, 0.1);
}
h2 {
   text-align: center;
   margin-bottom: 20px;
}
form {
   display: flex;
   flex-direction: column;
}
.form-field {
   display: flex;
   align-items: center;
   margin-bottom: 10px;
}
```

```
input[type="text"],
input[type="email"],
input[type="password"] {
    flex: 2;
    padding: 10px;
    border: 1px solid #ccc;
    border-radius: 4px;
    box-sizing: border-box;
}
input[type="submit"] {
    margin-top: 10px;
    margin-left: 25%;
    height: 45px;
    width: 50%;
    color: #fff;
    align-items: center;
    transition: all 0.2s ease;
    background: rgb(130, 106, 251);
}
.terms-condition {
    font-size: 0.9rem;
    margin-top: 10px;
}
```

OUTPUT:



EXPERIMENT-2

AIM: To explore Git and Github commands.

PROGRAM:

1)git config:

Usage: git config –global user.name "[name]"

Usage: git config –global user.email "[email address]"

This command sets the author name and email address respectively to be used with your commits.

2)git init:

Usage: git init [repository name]

This command is used to start a new repository.

3) git clone:

Usage: git clone [url]

This command is used to obtain a repository from an existing URL.

4) git status:

Usage: git status

This command lists all the files that have to be committed.

5) git add:

Usage: git add [file]

This command adds a file to the staging area.

Usage: git add *
This command adds one or more to the staging area.

6) git commit:

Usage: git commit -m "[Type in the commit message]"

This command records or snapshots the file permanently in the version history.

7) git log:

Usage: git log

This command is used to list the version history for the current branch.

8) git remote:

Usage: git remote add [variable name] [Remote Server Link]

This command is used to connect your local repository to the remote server.

9)git push:

Usage: git push [variable name] [branch]

This command sends the branch commits to your remote repository.

EXPERIMENT-3

AIM: To Practice Source code management on GitHub. Experiment with the source code written in experiment-2.

PROGRAM:

1) Create a new branch: Branches in Git allow developers to work on different features, fixes, or experiments in isolation from the main codebase, enabling parallel development and efficient collaboration.

>>git branch:

Usage: git branch

This command lists all the local branches in the current repository.

Usage: git branch [branch name]

This command creates a new branch.

Usage: git branch -d [branch name]

This command deletes the feature branch.

2) Switch to new branch:

>>git checkout:

Usage: git checkout [branch name]

This command is used to switch from one branch to another.

3) Make the changes in the code:

>>git diff:

Usage: git diff

This command shows the file differences which are not yet staged.

4) Track and commit changes:

5) Now merge the new branch with main branch to save changes in master branch:

>>git merge:

Usage: git merge [branch name]

This command merges the specified branch's history into the current branch.

6) Now push the changes into github repository:

7) Now you can see the changes in your github repository:

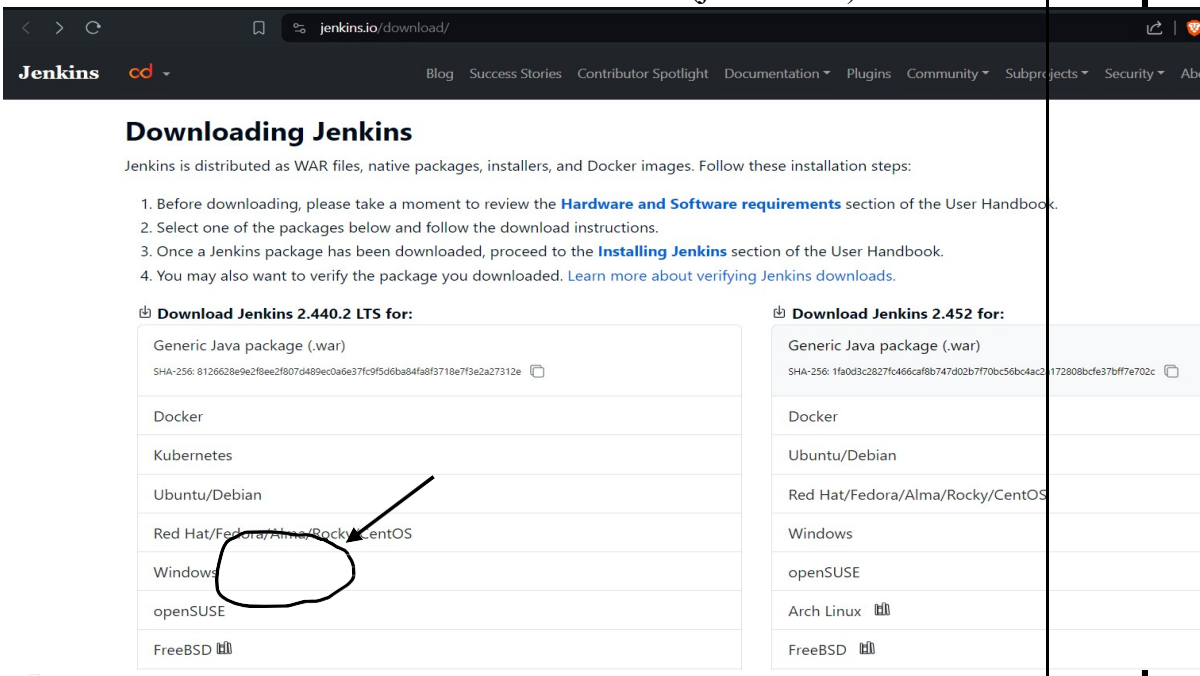
EXPERIMENT-4

Aim: Jenkins installation and setup, explore the environment.

PROCESS:

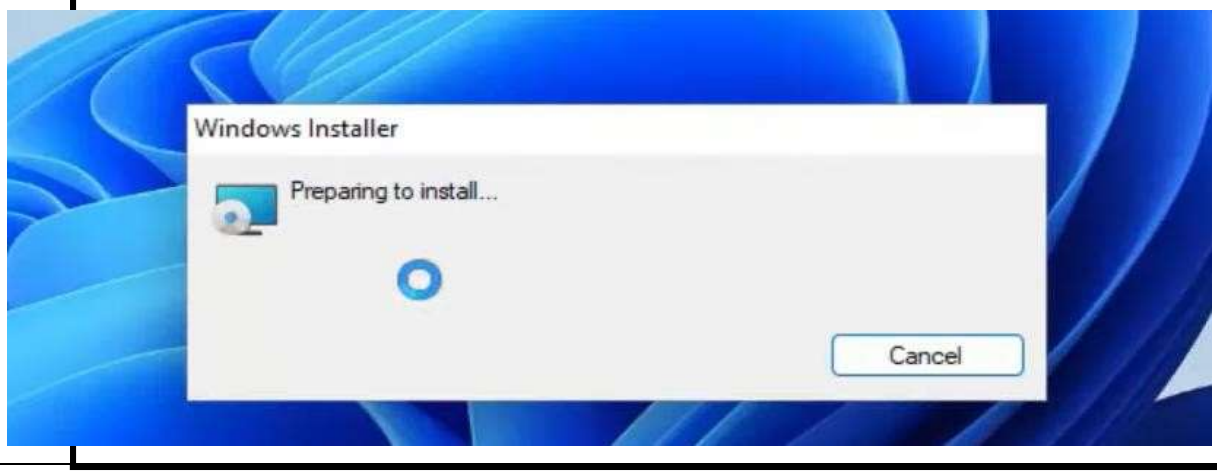
Step-1: Download Jenkins:-

- Go to the Jenkins website: https://www.jenkins.io/download/
- Click on the Windows link to download the Jenkins installer (jenkins.msi).



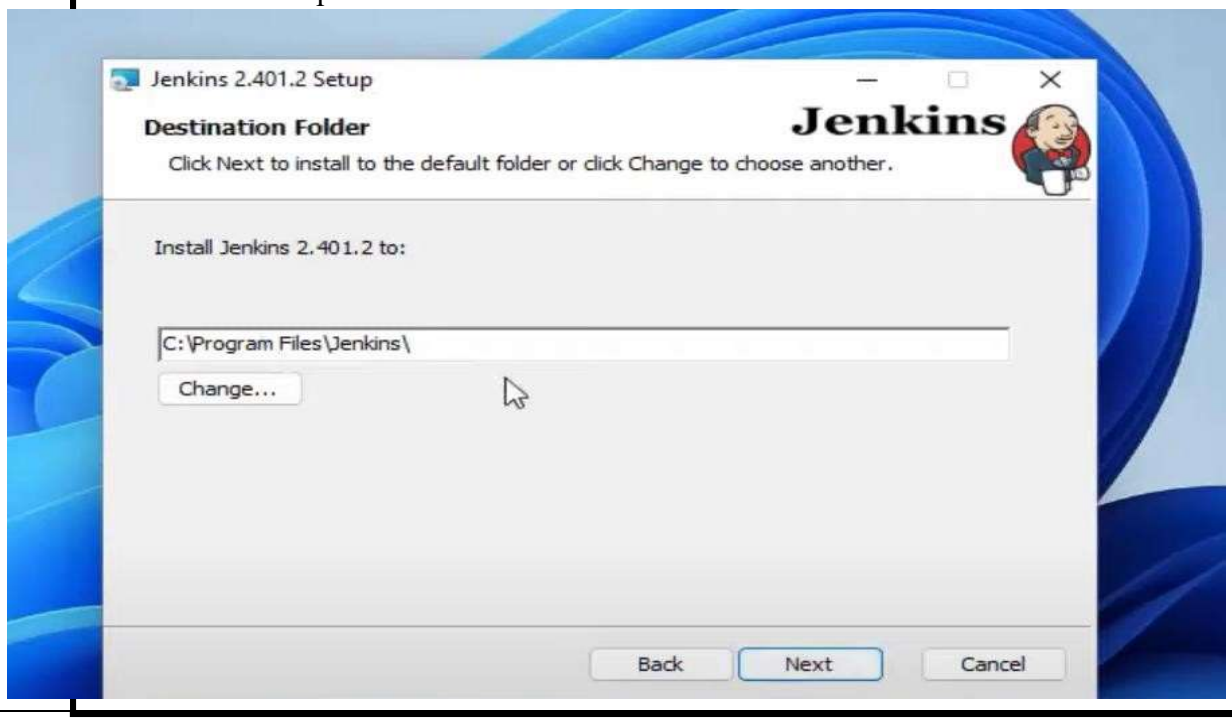
Step-2: Run the Installer:-

- Once the download is complete, locate the **jenkins.msi** file and double-click it to run the installer.
- If prompted by User Account Control, click "Yes" to allow the installer to make changes to your system.
- Follow the on-screen instructions in the Jenkins setup wizard.



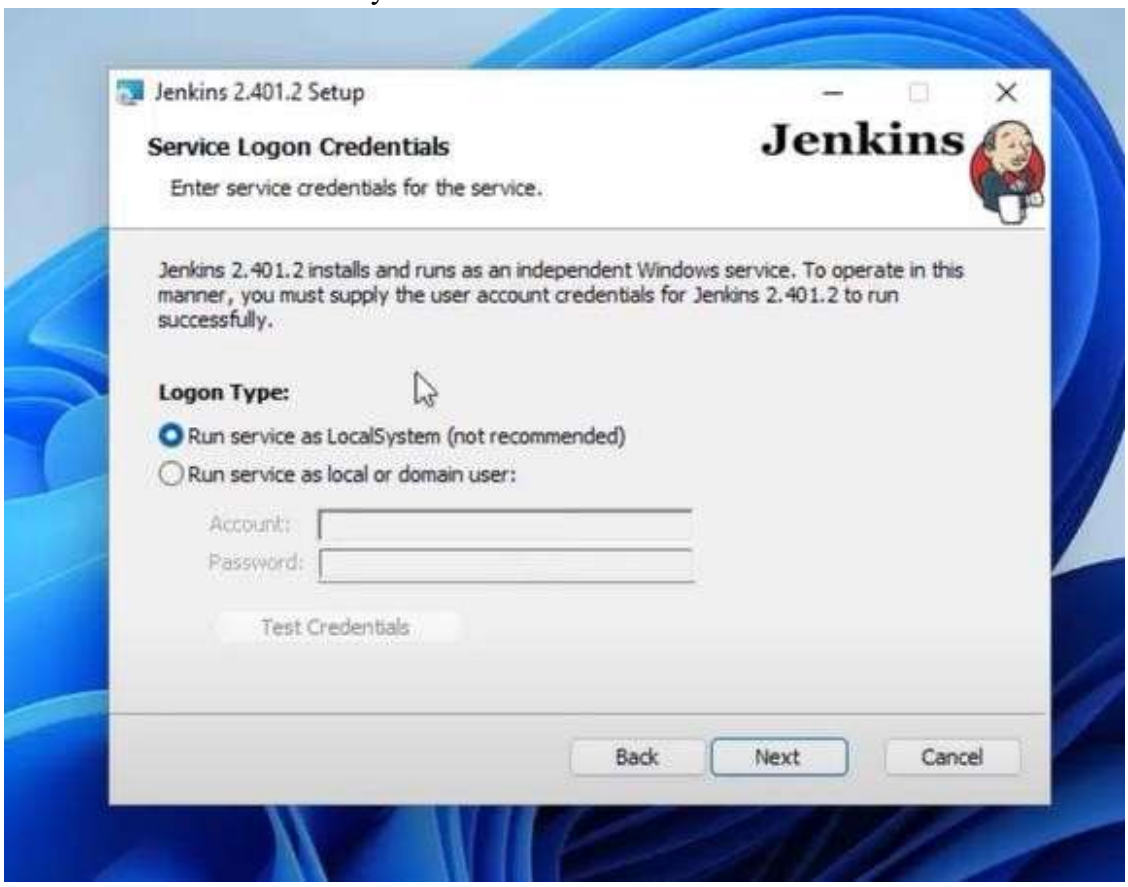
Step-3: Install Jenkins:

- Choose the installation directory for Jenkins. The default directory is usually C:\Program Files (x86)\Jenkins.
- Click "Next" to proceed with the installation.
- The installer will begin installing Jenkins on your system. This may take a few moments to complete.



Step-4:Service Login Credentials:

- Select “Run service as Localystem” and click on “Next”.

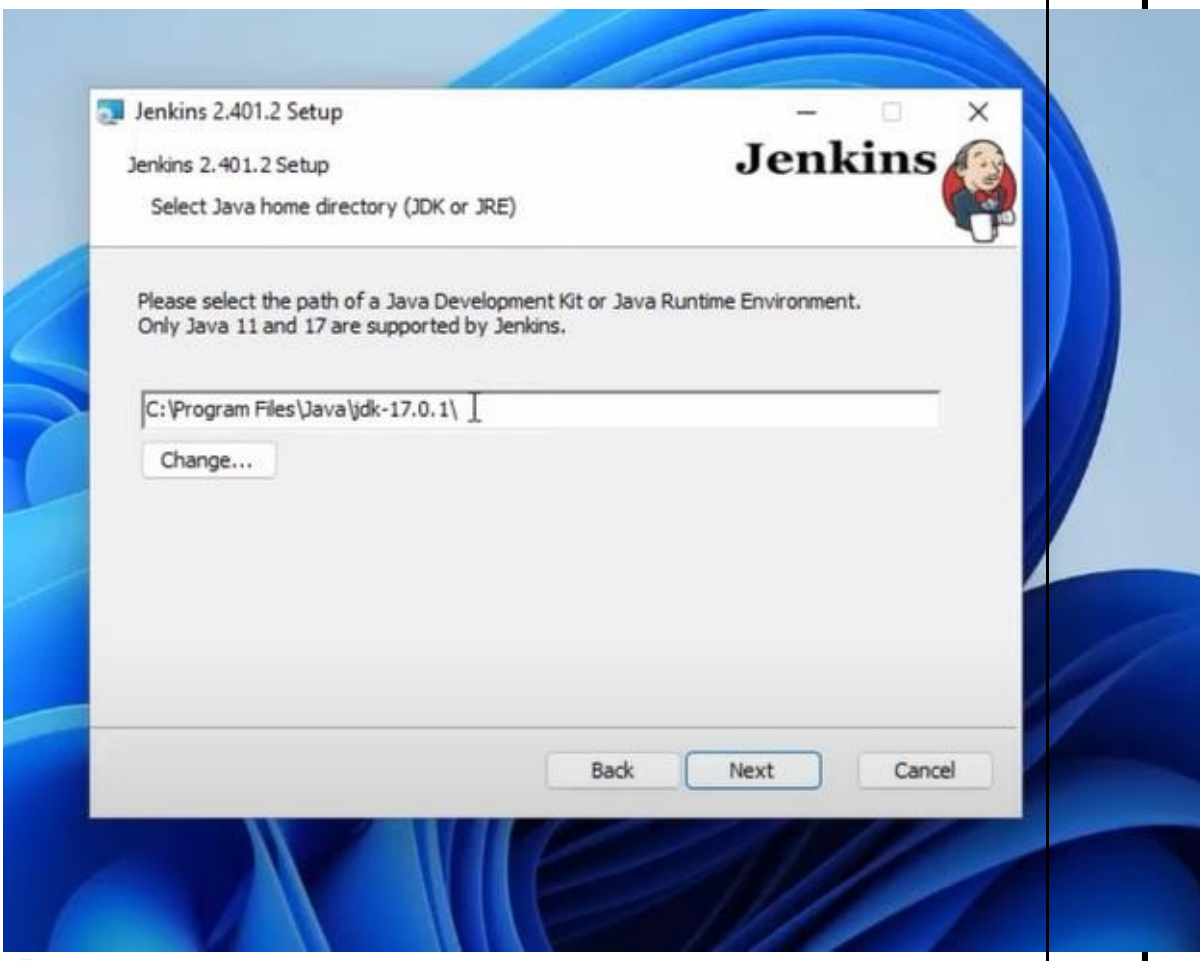


Step-5:Port Selection:

- Enter a port number ex: “8080” and test port .
- Click “Next”.

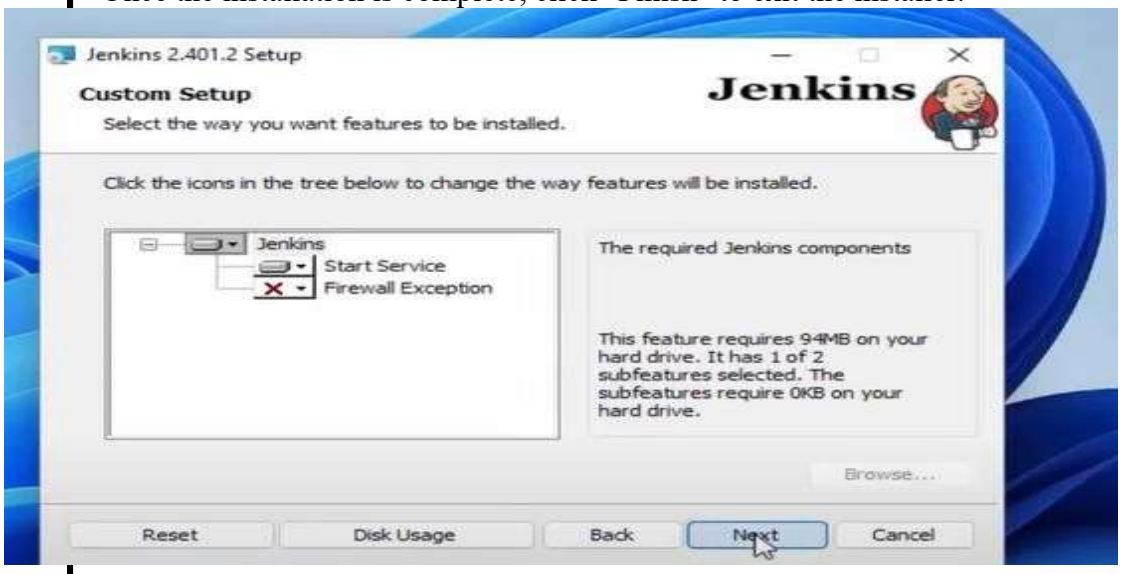
Step-6:Select JDK file directory:

- Jenkins automatically detect the directory path of JDK.
- If not then change path manually by clicking ” change ”.
- Click “Next”.

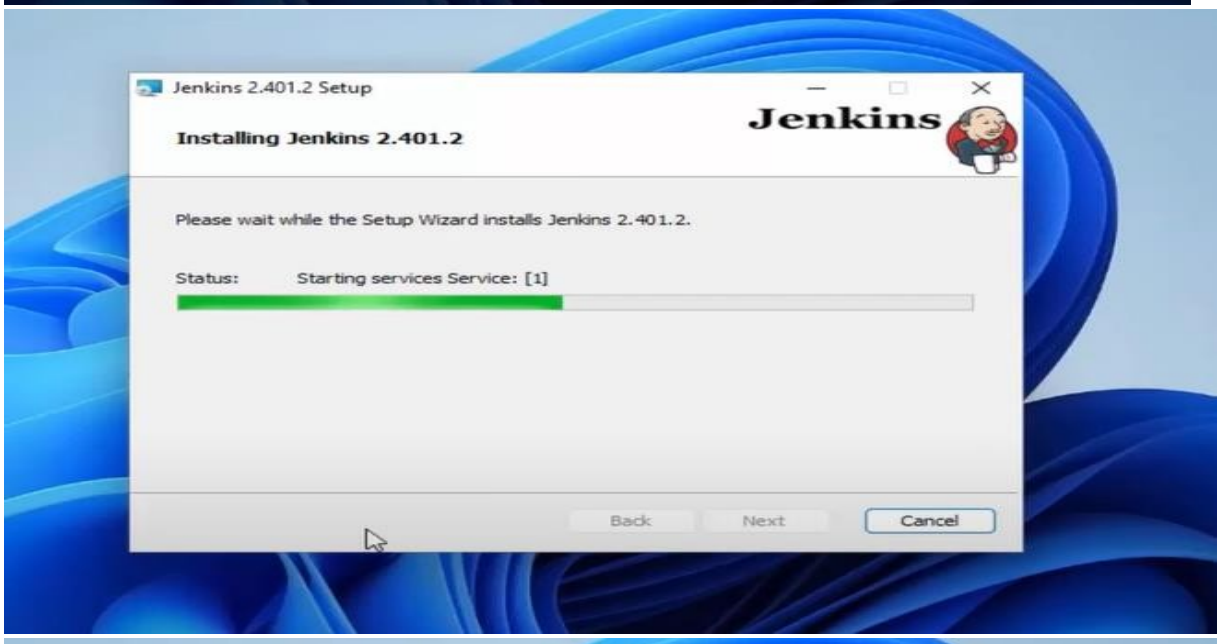


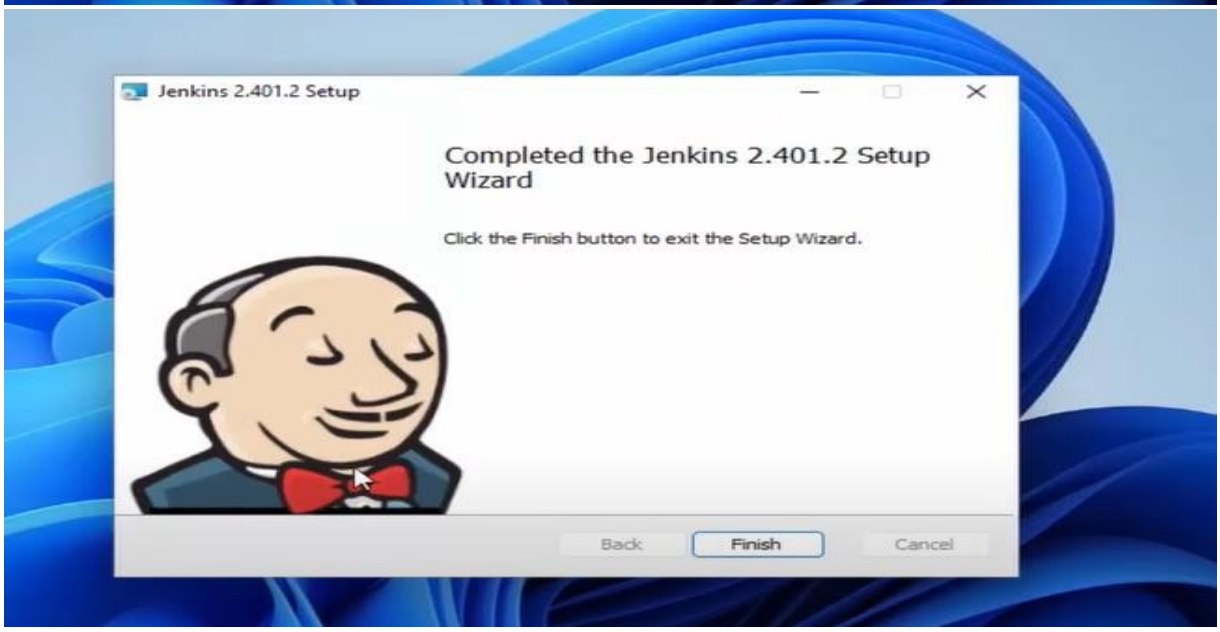
Step-7: Complete the Installation:

- Click “Next ” and then click “Install”.
- Once the installation is complete, click "Finish" to exit the installer.




User Account Control
Do you want to allow this app to make changes to your device?
Jenkins Automation Server 2.401.2
Verified publisher: CDF Binary Project a Series of LF Projects, LLC
File origin: Hard drive on this computer
Show more details
Yes
No


Jenkins 2.401.2 Setup
Installing Jenkins 2.401.2
Jenkins
Please wait while the Setup Wizard installs Jenkins 2.401.2.
Status: Starting services Service: [1]
Back
Next
Cancel


Jenkins 2.401.2 Setup
Completed the Jenkins 2.401.2 Setup Wizard
Click the Finish button to exit the Setup Wizard.
Back
Finish
Cancel

Step-8: Access Jenkins Web Interface:

- Open a web browser and navigate to http://localhost:8080 to access the Jenkins web interface.
- To obtain the initial administrator password, navigate to the location specified in the Jenkins setup wizard and open the initialAdminPassword file using a text editor.
- **C:\ProgramFiles(x86)\Jenkins\secrets\initialAdminPassword**).
- Copy the password from the file and paste it into the Jenkins setup wizard to unlock Jenkins.