

## SEPMI Experiment I

AIM-

To understand DevOps, principles, Practices and DevOps values and responsibilities

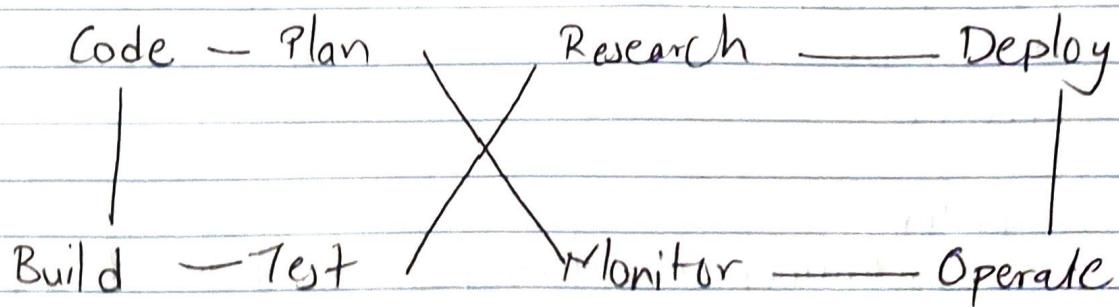
### Theory

DevOps is a combination of two words, Development and Operations. It is a culture to promote the development and operation process collectively.

Helps increase the organization's speed to deliver applications and services. Also allows orgs to serve their customers better and compete more strongly in the market.

DevOps can also be defined as a sequence of development and IT operations with better communication and collaboration.

DevOps has become one of the most valuable business disciplines for enterprises or organizations. With the help of DevOps quality and speed of the application delivery has improved to a greater extent.



### Build

Without devops the cost of consumption of the resources was evaluated based on the predefined individual usage with fixed hardware allocation.

Devops the usage of cloud sharing the resources come into the picture and the build is independent upon user needs which is a mechanism to control resources.

### Code

Many good practices such as Git enable the code to be used which ensures writing the code for business, helps to track changes, getting notified about the reason behind the difference in actual and expected code.

### Test

Application will be read for production after testing. In the case of manual testing it consumes more time in testing and moving the code to the output. The testing can be automated which decrease the time so that the time to deploy code can be reduced.

### Monitor

Continuous monitoring is used to identify any sort of failure. Also it helps in tracking the system accurately so that

## Advantages

DevOps is an excellent approach for quick development and deployment of applications

Responds faster to the market changes to improve business growth.

↓  
↓ Cost ↓

Escalate business profit by decreasing software delivery time and costs

## Disadvantages

DevOps professionals or expert developers are less available

Developing with DevOps is expensive  
Adoption of new technology is hard

### Deploy-

Many systems support the scheduler for automated deployment. The cloud management platform enables users to capture accurate insights and view the optimization scenario analysis on trends by the deployment of dashboards.

### Operate

Devops. Change the traditional approach of developing and testing separately. The team, operate in a collaborative way where both the teams actively participate throughout the service lifecycle. The operation team interacts with developers and they come up with a monitoring plan which serves business requirement.

### Release

Deployment to an environment can be done by automation. But when deployment is made to production environment, it is done by manual triggering. Many processes involved in release management commonly used to do the deployment.

### Principle

#### Collaboration

Data Based Decision Making

Customer Centric Decision Making

Constant Improvement

## EXPERIMENT 2

# TO UNDERSTAND VERSION CONTROL SYSTEM / SOURCE CODE MANAGEMENT, INSTALL GIT AND CREATE A GITHUB ACCOUNT

### THEORY:

#### What Is Version Control?

A version control system, also known as source control or revision control, tracks changes made to files over time, allowing users to retrieve specific versions when needed. While it can be applied in various scenarios, it is most commonly used in software development.

#### What is Git?

- Git is a free and open-source distributed version control system designed to handle projects of all sizes with speed and efficiency.
- It enables collaborative software development, allowing multiple developers to modify source code while tracking changes.
- Git was initially designed and developed by Linus Torvalds for Linux kernel development in 2005.
- Every Git working directory is a full-fledged repository with complete history and version tracking, independent of network access or a central server.
- Git enables teams to work on the same files simultaneously, resolving conflicts and maintaining version history.

#### What Is GitHub?

GitHub is a web-based platform that hosts software development projects and integrates with Git for version control. It provides a user-friendly interface and collaborative tools to enhance project management.

- It allows developers to create, manage, and access repositories remotely. ■ GitHub is essentially a collection of repositories that store project files and enable seamless collaboration.

### Use of Version Control Software

- Allows users to track different versions of a project and revert changes when necessary.
- Helps in comparing versions, debugging, and recovering lost data.

- Saves changes as patch files that can be applied to previous versions.
- Stores all versions on a central server, where developers can check out and update their work.

## Use Cases of GitHub

1. Version Control Allows developers to revert to previous versions when changes affect the project negatively.
2. Collaboration & Code Review Enables teams to review each other's code, improving productivity and efficiency.
3. Issue Tracking Developers can assign issues to team members and track progress.
4. Open-Source Development GitHub is widely used for open-source projects, allowing public contributions.

## Characteristics of Git

1. Strong Support for Non-Linear Development
  - Supports rapid branching and merging.
  - Changes are merged frequently.
  - Lightweight branches make development faster.
2. Distributed Development
  - Every developer gets a local copy of the repository.
  - Changes can be merged efficiently between repositories.
3. Compatibility with Existing Systems
  - Git supports CVS server emulation, allowing existing CVS clients to access Git repositories.
4. Efficient Handling of Large Projects
  - Faster and more scalable than other version control systems.
  - Fetching data from a local repository is quicker than from a remote server.
5. Data Assurance
  - Git stores history in a way that ensures integrity—old versions cannot be changed unnoticed.
6. Automatic Garbage Collection
  - Git performs automatic garbage collection when enough loose objects are created.
  - Can be explicitly triggered using `git gc --prune`.
7. Periodic Explicit Object Packing
  - Git stores objects as separate files, later compressed into a packfile.
  - This improves performance but is computationally expensive.

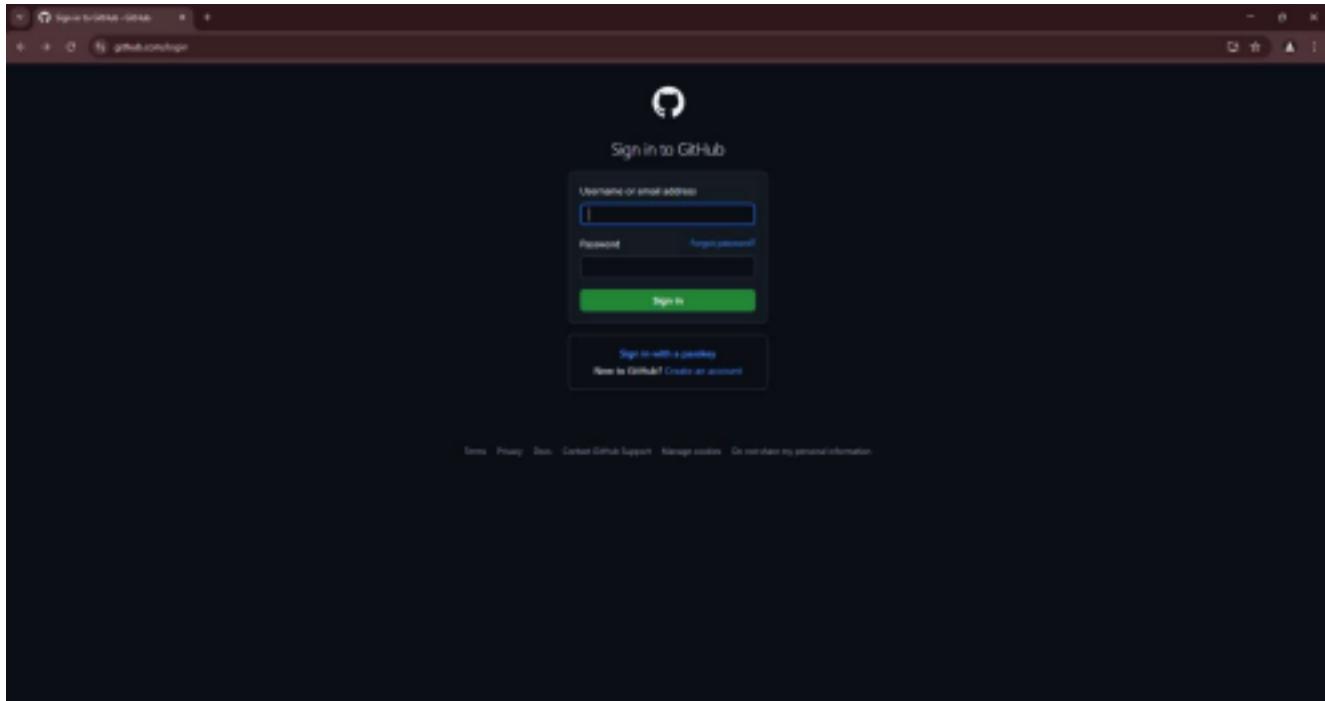
## How Does Git Work?

■ A Git repository is a key-value store where all objects are indexed by their SHA-1 hash.

- Objects include commits, files, tags, and tree nodes.
- Git works by taking snapshots of files at different points in time.

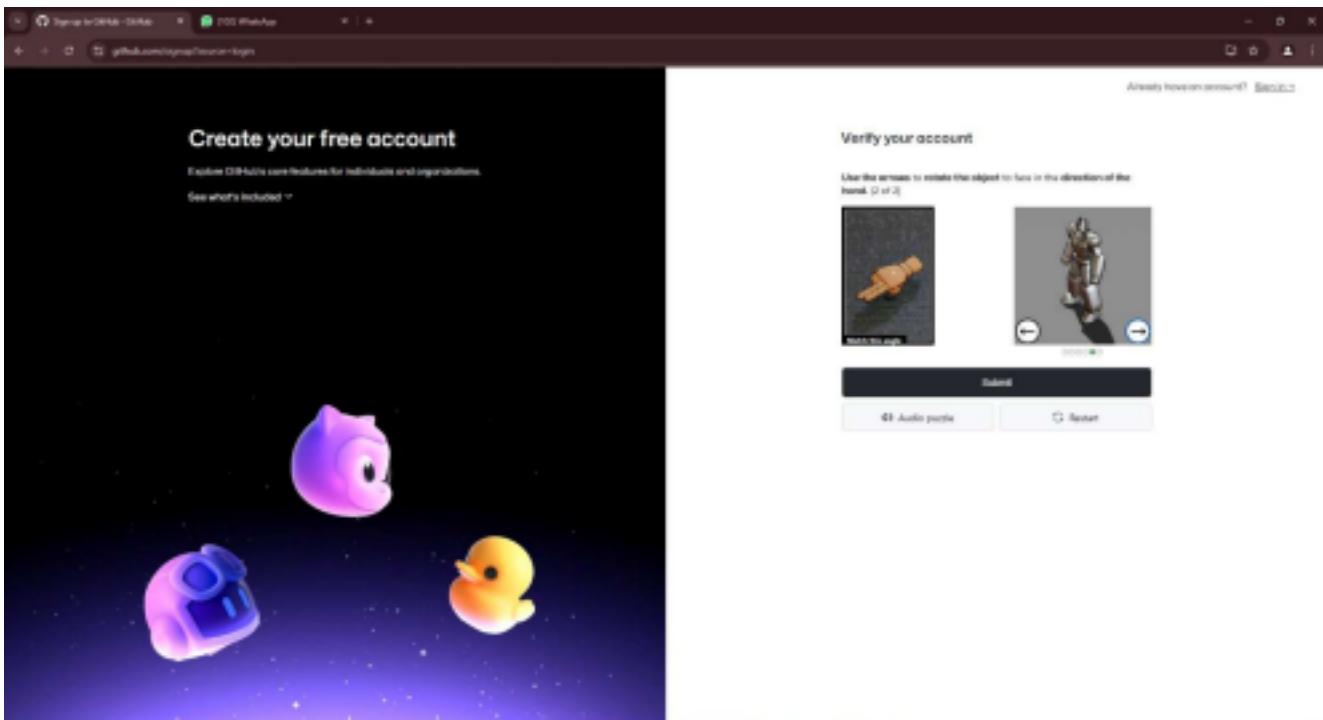
## Steps to Creating a GitHub Account

1. Open a web browser and go to [GitHub Signup](https://github.com/signup).



2. Enter a unique username. Provide a valid email address. Create a strong password.

3. Click "Create account". Complete the CAPTCHA verification.



4. Choose your email preferences and click "Continue".
5. Verify your email by clicking the link sent to your registered email address.
6. Set up your profile (optional) and Select Free plan

The GitHub dashboard for user 'rya23' displays the following information:

- Top repositories:** TeamMKA/HackDesk\_Nerd.js, TeamMKA/AugenBlick\_2025, TeamMKA/RUBIX25\_27\_NUCLITRON, rya23/RUBIX25\_27\_NUCLITRON, rya23/HackSync\_BetterCallDevs, rya23/fastapi, rya23/flask\_crud.
- Your teams:** TeamMKA/bitbusters.
- Recent activity:**
  - manaspatil7 created a repository 2 hours ago: manaspatil7/Great-Ninja-Hack
  - lucidrains starred a repository 2 days ago: cheatface/nimcrypto (Nim cryptographic library)
- Trending repositories:** Trending repositories (See more), ourongxing/newsnow (Elegant reading of real-time and hottest news).
- Latest changes:**
  - 14 hours ago: Deprecation of cvss field in security advisories API
  - 15 hours ago: Repository ownership limits
  - 16 hours ago: GPT-4o Copilot: Your new code completion model is now generally available
  - 2 days ago: Transitive dependencies are now available for Maven
- Explore repositories:**
  - PufferAI / PufferLib: Simplifying reinforcement learning for complex game environments (1.8k stars)
  - amueller / introduction\_to\_ml\_with\_python: Notebooks and code for the book "Introduction to Machine Learning with Python"

7. Your GitHub account is now ready to use!

## Steps to Create a Repository on GitHub

1. Log in to GitHub:
  - o Go to [GitHub](#) and sign in to your account.

## 2. Navigate to Repositories:

- o Click on your profile icon (top-right corner) and select "Your repositories" from the dropdown.
- o Alternatively, click on the "+" icon in the top-right corner and select "New repository".

The screenshot shows a dark-themed dashboard with the following sections:

- Top repositories:** A list of repositories owned by the user, including TeamMKA/HackDesk\_Nerd.js, TeamMKA/AugenBlick\_2025, TeamMKA/RUBIX25\_27\_NUCLITRON, rya23/RUBIX25\_27\_NUCLITRON, rya23/HackSync\_BetterCallDevs, rya23/fastapi, and rya23/flask\_crud.
- Ask Copilot:** A search bar with placeholder text "Type ⌘ to search" and a "Copilot" icon.
- Home:** A feed of recent events:
  - manaspalit7 created a repository 2 hours ago
  - manaspalit7/Great-Ninja-Hack
  - lucidrains starred a repository 2 days ago
  - cheatfate/nimcrypto: Nim cryptographic library (1.8k stars)
- Latest changes:** A list of recent repository updates:
  - Deprecation of cvss field in security advisories API (14 hours ago)
  - Repository ownership limits (15 hours ago)
  - GPT-4o Copilot: Your new code completion model is now generally available (16 hours ago)
  - Transitive dependencies are now available for Maven (2 days ago)
- Explore repositories:** A section showing trending repositories:
  - PufferAI / PufferLib: Simplifying reinforcement learning for complex game environments (1.8k stars)
  - amueller / introduction\_to\_ml\_with\_python: Notebooks and code for the book "Introduction to Machine Learning with Python"

## 3. Create a New Repository:

- o Enter a repository name (e.g., my-project).
- o Optionally, add a description.
- o Choose the visibility:
  - Public (anyone can see your repository).
  - Private (only you and collaborators can access it).

## 4. Initialize the Repository (Optional):

- o You can check "Add a README file" to include a basic introduction.
- o Optionally, add a .gitignore file for excluding certain files from tracking.
- o Choose a license if needed.

## 5. Create the Repository:

- o Click the "Create repository" button.

**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 (\*).

**Repository template**

No template

Start your repository with a template repository's contents.

**Owner \*** rya23 / **Repository name \*** sepm\_sem6  
sepm\_sem6 is available.

Great repository names are short and memorable. Need inspiration? How about [musical-carnival](#) ?

**Description (optional)** sepm sem 6 repo

**Visibility** 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](#).

## 6. Set Up Locally (Optional):

- o Copy the repository URL and use Git to clone it:
- o `git clone https://github.com/your-username/your-repository.git`
- o Navigate to the folder and start working on your project!

Now your repository is ready to use! ♦♦♦

rya23 / sepm\_sem6

**Code** Issues Pull requests Actions Projects Wiki Security Insights Settings

**Code** About

sepm\_sem6 (Public)

main 1 Branch 0 Tags

rya23 Initial commit Tab4f29 · now 1 Commit

README.md Initial commit now

README

**sepm\_sem6**

sepm sem 6 repo

sepm sem 6 repo

0 stars 0 forks 0 watching 0 releases published Create a new release

No packages published Publish your first package

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

## EXPERIMENT – 3

AIM - To Perform various GIT operations on local and Remote repositories using GIT

### Cheat-Sheet

```
15L@203-06 MINGW64 ~/Desktop/aarya-git (master) $  
git config --global user.name "aarya"
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git (master)  
$ git config --global user.email aarya48.shah@gmail.com
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git (master)  
$ git config --global --list  
user.name=aarya  
user.email=aarya48@gmail.com
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git (master)  
$ mkdir git-demo-project
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git (master)  
$ cd git-demo-project
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git init  
Initialized empty Git repository in C:/Users/15L/Desktop/aarya-git/git-demo-project/.git/
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ ls -a  
./ ../.git/
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ rm -rf .git/
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ rm -rf .git
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ ls -al  
total 0  
drwxr-xr-x 1 15L 197121 0 Jan 23 13:40 ./  
drwxr-xr-x 1 15L 197121 0 Jan 23 13:39 ../
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git init
```

```
Initialized empty Git repository in C:/Users/15L/Desktop/aarya-git/git-demo-project/.git/
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git add .
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git status
On branch master
```

```
No commits yet
```

```
Changes to be committed:
(use "git rm --cached <file>..." to unstage)
  new file:   sample.txt
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git commit -m "first commit"
[master (root-commit) d489bcf] first commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 sample.txt
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    index.html
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git add .
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   index.html
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git commit -am "express commit"
[master bef012b] express commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.html
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master) $  
git status  
On branch master  
nothing to commit, working tree clean
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git log  
commit bef012be210dc3856730420ce2228425d43e58f5 (HEAD -> master)  
Author: aarya <aarya48.shah@gmail.com>  
Date:   Tue Jan 23 13:43:06 2024 +0530
```

express commit

commit

```
d489bcf61da3943db131c6d0274d9ear00737319  
Date:   Tue Jan 23 13:41:06 2024 +0530  
Author: aarya <aarya48.shah@gmail.com>
```

first commit

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ nano index.html
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git status  
On branch master  
Changes not staged for commit:  
  (use "git add <file>..." to update what will be committed)  
  (use "git restore <file>..." to discard changes in working directory)  
    modified:   index.html
```

no changes added to commit (use "git add" and/or "git commit -a")

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git log  
commit bef012be210dc3856730420ce2228425d43e58f5 (HEAD -> master)  
Author: aarya <aarya48.shah@gmail.com>  
Date:   Tue Jan 23 13:43:06 2024 +0530
```

express commit

commit

```
d489bcf61da3943db131c6d0274d9ear00737319  
Date:   Tue Jan 23 13:41:06 2024 +0530  
Author: aarya <aarya48.shah@gmail.com>
```

first commit

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
```

```
$ touch teststatus

15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)

    modified:   index.html
```

```
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    teststatus

no changes added to commit (use "git add" and/or "git commit -a")
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git checkout -- teststatus
error: pathspec 'teststatus' did not match any file(s) known to git
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git checkout --index.html
error: unknown option `index.html'
usage: git checkout [<options>] <branch>
  or: git checkout [<options>] [<branch>] -- <file>...
      --recurse-submodules[=<checkout>]
      --progress
      -m, --merge
      --conflict <style>
      -d, --detach
      -t, --track[=(direct|inherit)]
      --force
      --orphan <new-branch>
      --overwrite-ignore
      --ignore-other-worktrees
      -2, --ours
      -3, --theirs

      create and checkout a new branch
      create/reset and checkout a branch
      create reflog for new branch
      second guess 'git checkout <no-such-branch>' (default)
      use overlay mode (default)
      suppress progress reporting
      control recursive updating of submodules
      force progress reporting
      perform a 3-way merge with the new branch
      conflict style (merge, diff3, or zdiff3)
      detach HEAD at named commit
      set branch tracking configuration
      force checkout (throw away local modifications)
      new unparented branch
      update ignored files (default)
      do not check if another worktree is holding the given ref
      checkout our version for unmerged files
      checkout their version for unmerged files
```

```
-p, --patch           select hunks interactively
--ignore-skip-worktree-bits
                      do not limit pathspecs to sparse entries only
--pathspec-from-file <file>
                      read pathspec from file
--pathspec-file-nul
                      with --pathspec-from-file, pathspec elements are separated
with NUL character
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git add index.html
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next
time Git touches it
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   index.html
```

```
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    teststatus
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git add teststatus
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   index.html
    new file:   teststatus
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git commit -am "express commit"
[master 298a0e6] express commit
 2 files changed, 1 insertion(+)
 create mode 100644 teststatus
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git status
On branch master
nothing to commit, working tree clean
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master) $  
git log commit 298a0e6689dd8b9d98ebc4e5413441e244b1036d (HEAD ->  
master) Author: aarya <aarya48.shah@gmail.com>
```

Date: Tue Jan 23 13:48:37 2024 +0530

express commit

commit

```
bef012be210dc38567304206e2228425d43e58f5  
Date: Tue Jan 23 13:43:06 2024 +0530  
Author: aarya <aarya48.shah@gmail.com>
```

express commit

commit

```
d489bcf61da3943db131c6d0274d9eac00737319  
Date: Tue Jan 23 13:41:06 2024 +0530  
Author: aarya <aarya48.shah@gmail.com>
```

first commit

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git log --oneline  
298a0e6 (HEAD -> master) express commit  
bef012b express commit  
d489bcf first commit
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git remote add origin https://github.com/aarya-tsec/SEPM.git
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ rm -rf ^C
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ rm -rf https://github.com/aarya-tsec/SEPM.git
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git remote add origin https://github.com/aarya-tsec/EXP3.git  
error: remote origin already exists.
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git origin  
git: 'origin' is not a git command. See 'git --help'.
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)  
$ git show origin  
fatal: ambiguous argument 'origin': unknown revision or path not in the working tree.  
Use '--' to separate paths from revisions, like this:
```

```
'git <command> [<revision>...] -- [<file>...]'
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git remote show origin
* remote origin
  Fetch URL: https://github.com/aarya-tsec/SEPM.git
  Push URL: https://github.com/aarya-tsec/SEPM.git
  HEAD branch: (unknown)
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git remote remove origin
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git remote show origin
fatal: 'origin' does not appear to be a git repository
fatal: Could not read from remote repository.
```

Please make sure you have the correct access rights  
and the repository exists.

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git remote add origin https://github.com/aarya-tsec/EXP3.git
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git remote show origin
* remote origin
  Fetch URL: https://github.com/aarya-tsec/EXP3.git
  Push URL: https://github.com/aarya-tsec/EXP3.git
  HEAD branch: (unknown)
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (master)
$ git branch -M main
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (main)
$ git push -u origin main
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 20 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 691 bytes | 691.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/aarya-tsec/EXP3.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (main)
$ git pull
Already up to date.
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (main) $  
git pull remote: Enumerating objects: 5, done. remote: Counting  
objects: 100% (5/5), done. remote: Compressing objects: 100% (2/2),  
done. remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
Unpacking objects: 100% (3/3), 957 bytes | 239.00 KiB/s, done. From  
https://github.com/aarya-tsec/EXP3
```

```
298a0e6..7242627 main          -> origin/main  
Updating 298a0e6..7242627  
Fast-forward  
 index.html | 2 +-  
 1 file changed, 1 insertion(+), 1 deletion(-)
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (main)  
$ git log --oneline origin/main  
7242627 (HEAD -> main, origin/main) Update index.html  
298a0e6 express commit  
bef012b express commit  
d489bcf first commit
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (main)  
$ $ git fetch  
bash: $: command not found
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (main)  
$ git fetch
```

```
15L@203-06 MINGW64 ~/Desktop/aarya-git/git-demo-project (main)  
$ git merge origin/main  
Already up to date.
```

```

MINGW64:/Users/13U/git-dvcs/git-dem-project
PS C:\Users\Lab805_3\Desktop\sepm>
PS C:\Users\Lab805_3\Desktop\sepm> git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
PS C:\Users\Lab805_3\Desktop\sepm> git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .txt

nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\Lab805_3\Desktop\sepm> git add *
PS C:\Users\Lab805_3\Desktop\sepm> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:  .txt

PS C:\Users\Lab805_3\Desktop\sepm> git add -m "first commit"
  --mirror=(push|fetch)
      set up remote as a mirror to push to or fetch from

15L8203-010 MINGW64 ~/git-dvcs/git-dem-project (master)
$ git remote rm origin
15L8203-010 MINGW64 ~/git-dvcs/git-dem-project (master)
$ git push -u origin master
fatal: 'origin' does not appear to be a git repository
fatal: could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.

15L8203-010 MINGW64 ~/git-dvcs/git-dem-project (master)
$ git pull
remote: no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

MINGW64:/Users/13U/git-dvcs/git-dem-project
$ git config --global user.name "Aditya"
15L8203-010 MINGW64 ~/git-dvcs (master)
$ git config --global user.email "adikonda@gmail.com"
15L8203-010 MINGW64 ~/git-dvcs (master)
$ git config --global --list
color.ui=true
user.name=Aditya
user.email=adikonda@gmail.com

15L8203-010 MINGW64 ~/git-dvcs (master)
$ mkdir git-demo-project
mkdir: cannot create directory 'git-demo-project': File exists

15L8203-010 MINGW64 ~/git-dvcs (master)
$ rm git-demo-project
rm: cannot remove 'git-demo-project': Is a directory

15L8203-010 MINGW64 ~/git-dvcs (master)
$ rmdir git-demo-project
rmdir: failed to remove "git-demo-project": Directory not empty

15L8203-010 MINGW64 ~/git-dvcs (master)
$ rmdir -f git-demo-project
rmdir: unknown option -- f
Try 'rmdir --help' for more information.

15L8203-010 MINGW64 ~/git-dvcs (master)
$ rmdir --help
Usage: rmdir [OPTION]... DIRECTORY...
Remove the DIRECTORY(es), if they are empty.

--ignore-fail-on-non-empty
      ignore each failure that is solely because a directory
      is non-empty
-p, --parents  remove DIRECTORY and its ancestors; e.g., 'rmdir -p a/b/c' is
              similar to 'rmdir a/b/c a/b/a'
-v, --verbose   output a diagnostic for every directory processed
--help         display this help and exit
--version      output version information and exit

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Report any translation bugs to <https://translationproject.org/team/>
Full documentation <https://www.gnu.org/software/coreutils/rmdir>
or available locally via: info '(coreutils) rmdir invocation'

15L8203-010 MINGW64 ~/git-dvcs (master)
$ rm -rf git-demo-project

15L8203-010 MINGW64 ~/git-dvcs (master)

```

```
PS C:\Users\101\git-dvcs\git-dem-project
$ git pull -u origin master
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

git pull <remote> <branch>
If you wish to set tracking information for this branch you can do so with:
  git branch --set-upstream-to=<remote>/<branch> master

15L8203-010 MINGW64 ~/git-dvcs/git-dem-project (master)
$ git push -u origin master
fatal: 'origin' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.

15L8203-010 MINGW64 ~/git-dvcs/git-dem-project (master)
$ git remote show origin
fatal: 'origin' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.

15L8203-010 MINGW64 ~/git-dvcs/git-dem-project (master)
$ git remote add origin https://github.com/Adityadikonda/ISRO.git
15L8203-010 MINGW64 ~/git-dvcs/git-dem-project (master)
$ git remote show origin
* remote origin
  Fetch URL: https://github.com/Adityadikonda/ISRO.git
  Push URL: https://github.com/Adityadikonda/ISRO.git
  HEAD branch: main
  Remote branch:
    main new (next fetch will store in remotes/origin)

15L8203-010 MINGW64 ~/git-dvcs/git-dem-project (master)
$ git pull
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 26 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (26/26), 461.85 KiB | 1.68 MiB/s, done.
From https://github.com/Adityadikonda/ISRO
 * [new branch]  main      -> origin/main
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.
```

## CONCLUSION:

Thus, we have successfully studied and performed various GIT operations on local and Remote repositories using GIT Cheat-Sheet.

## **Software Engineering & Project Management Lab Experiment No: - 04**

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

### **Theory:**

Continuous Integration (CI) is a DevOps practice where code changes are automatically built, tested, and integrated into a shared repository multiple times a day. It helps in early detection of errors, reduces integration problems, and improves software quality.

### **Jenkins: An Overview**

Jenkins is an open-source CI/CD automation tool used for building, testing, and deploying applications. It allows developers to automate software development workflows and ensures a seamless integration process. Jenkins supports various build tools like **Maven**, **Ant**, and **Gradle** to compile and package applications.

### **Installing and Configuring Jenkins**

#### **1. Download and Install Jenkins**

- o Install Java (JDK) as a prerequisite.
- o Download Jenkins from the official website and install it on the server.
- o Start Jenkins and configure initial setup using an administrator password.

#### **2. Installing Build Tools**

- o Install **Maven**, **Ant**, or **Gradle** depending on project requirements.
- o Configure Jenkins to recognize the installed build tool.

#### **3. Creating a Build Job in Jenkins**

- o Navigate to **Jenkins Dashboard** → **New Item** → **Freestyle Project/Pipeline**.
- o Configure the **Git repository URL** to fetch the source code.
- o Select the **Build Tool (Maven/Ant/Gradle)** and define the build command.
- o Set up triggers (e.g., Git webhooks) for automatic build execution.
- o Save and trigger the build job to verify the setup.

## **Software Engineering & Project Management Lab Experiment No: - 04**

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

To install Jenkins following software packages are required:

- 1) GIT ([git-scm.com](http://git-scm.com))
- 2) Notepad++ (<https://notepad-plus-plus.org/downloads/>)
- 3) Latest Java development kit (JDK)
- 4) Jenkins
- 5) Apache Maven (Optional)

Step 1:- Install GIT

Step 2 :- Install Notepad++

Step 3 :- Install Java

Step 4 :- Install Jenkins

Step 5 :- Install Maven

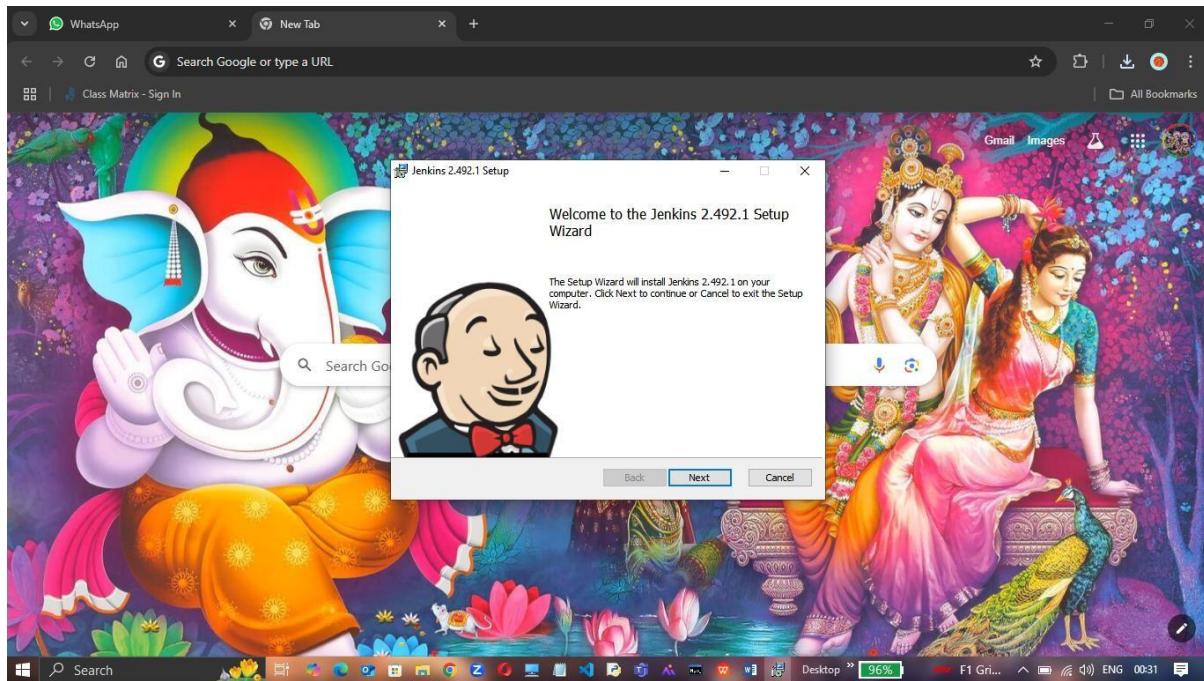
Jenkins is an open source automation tool written in Java with plugins built for Continuous Integration purpose. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies.

**Step 1:-** Open <https://www.jenkins.io/doc/book/installing/windows/> and install Jenkins.

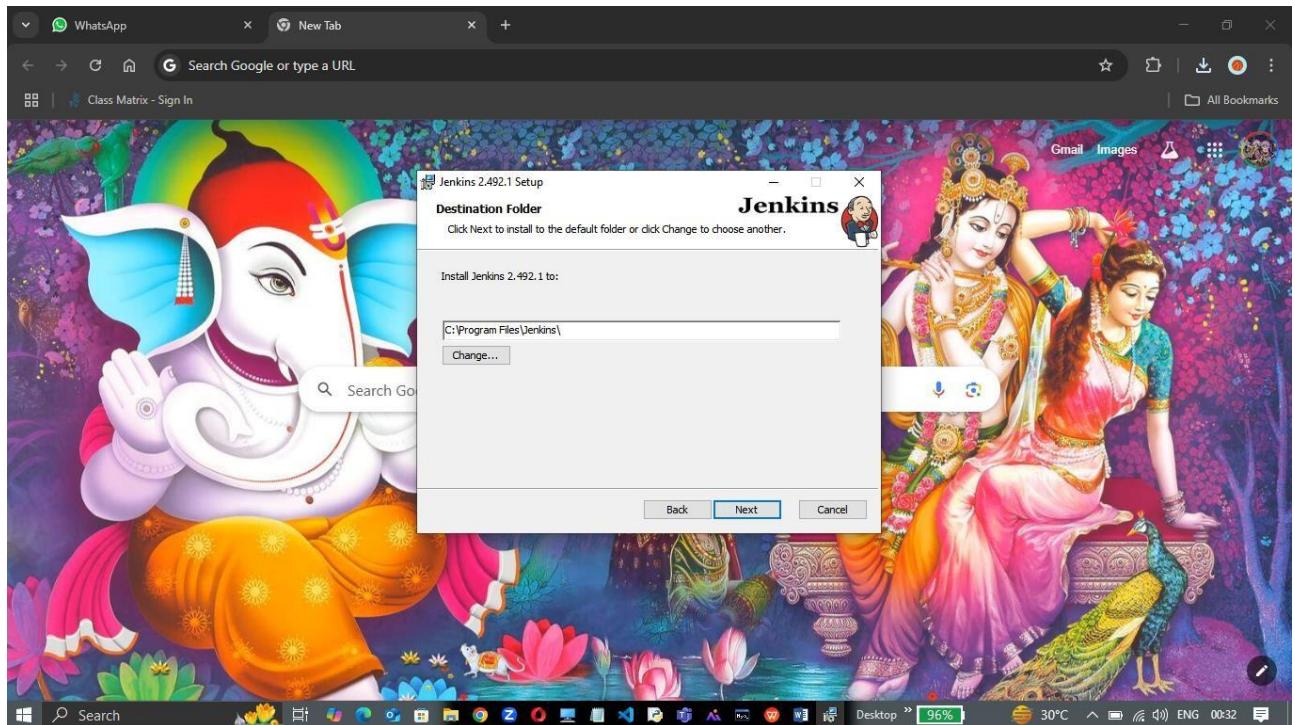
Open the installed .exe setup

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**



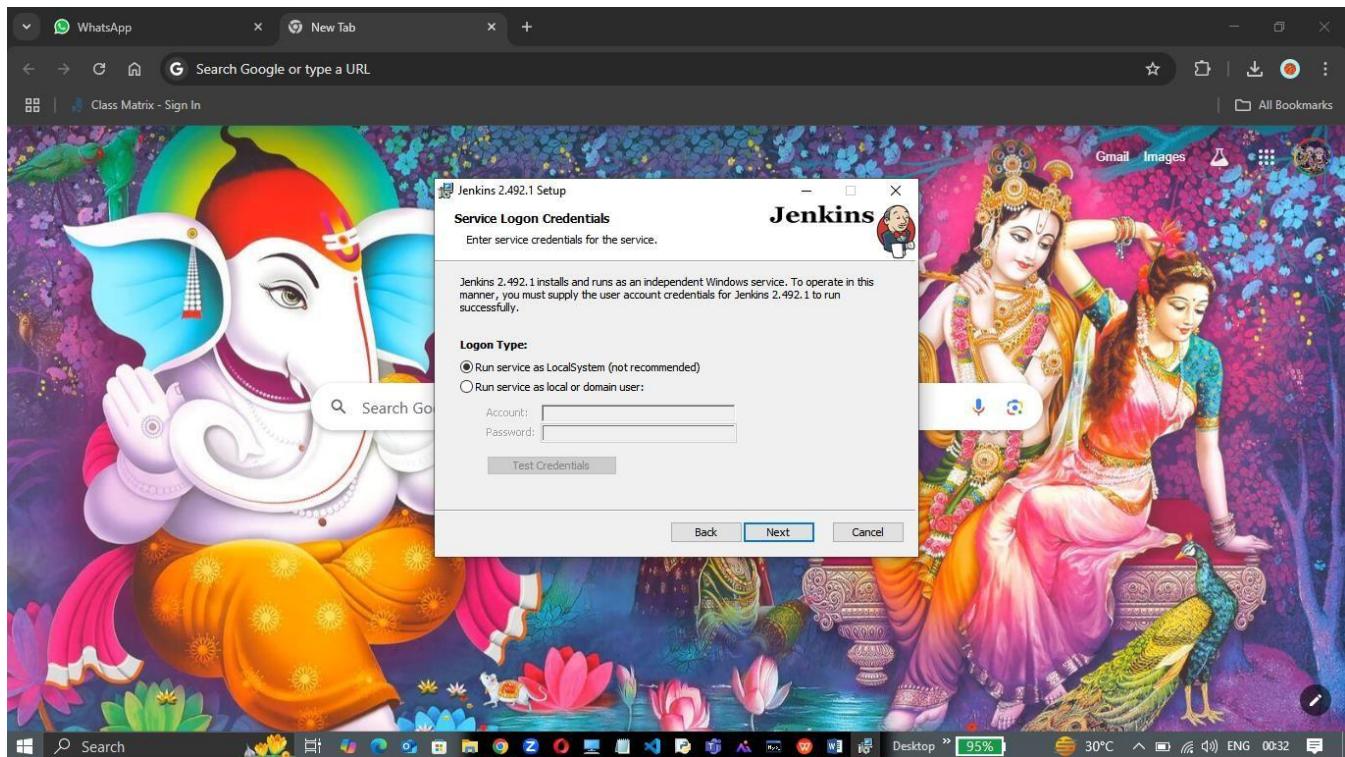
**Step 2:** Locate the folder where you want to install Jenkins in the location path:



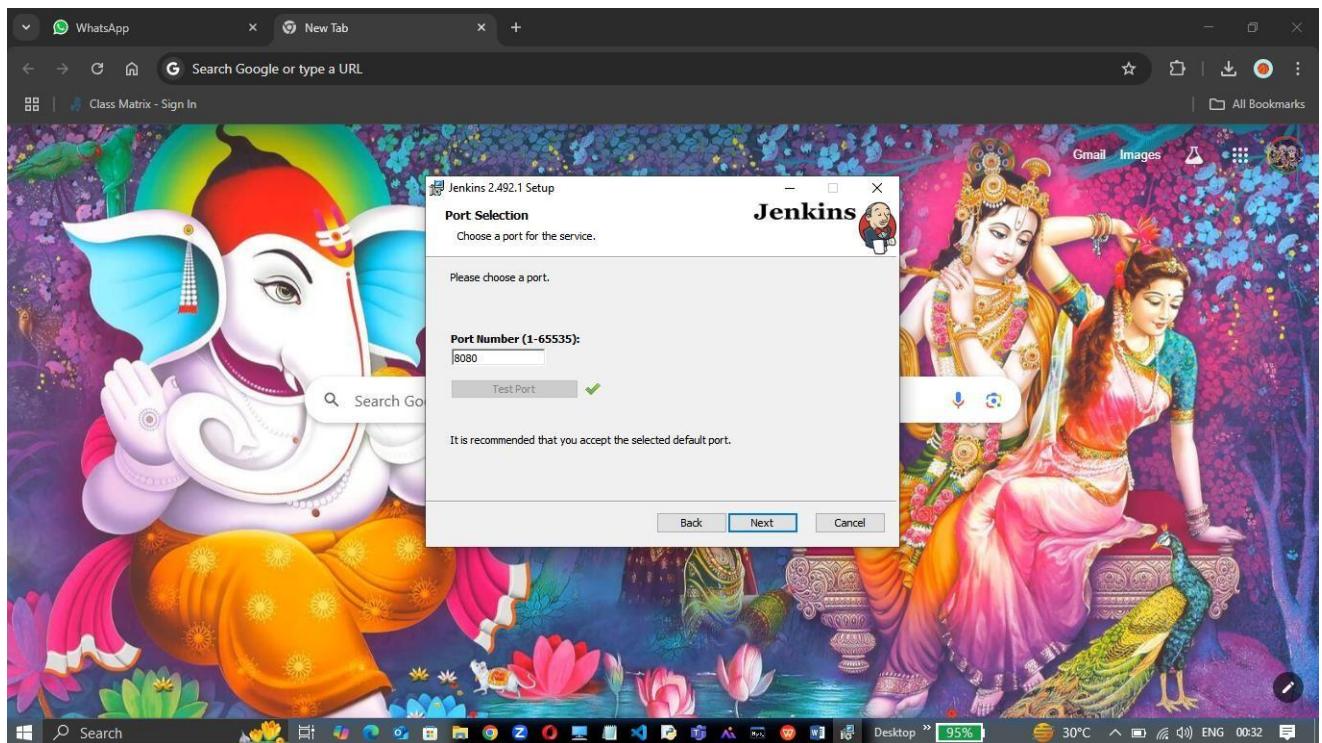
## Software Engineering & Project Management Lab Experiment No: - 04

**Aim:** To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job

**Step 3:** Select service as Local System and proceed to Next.



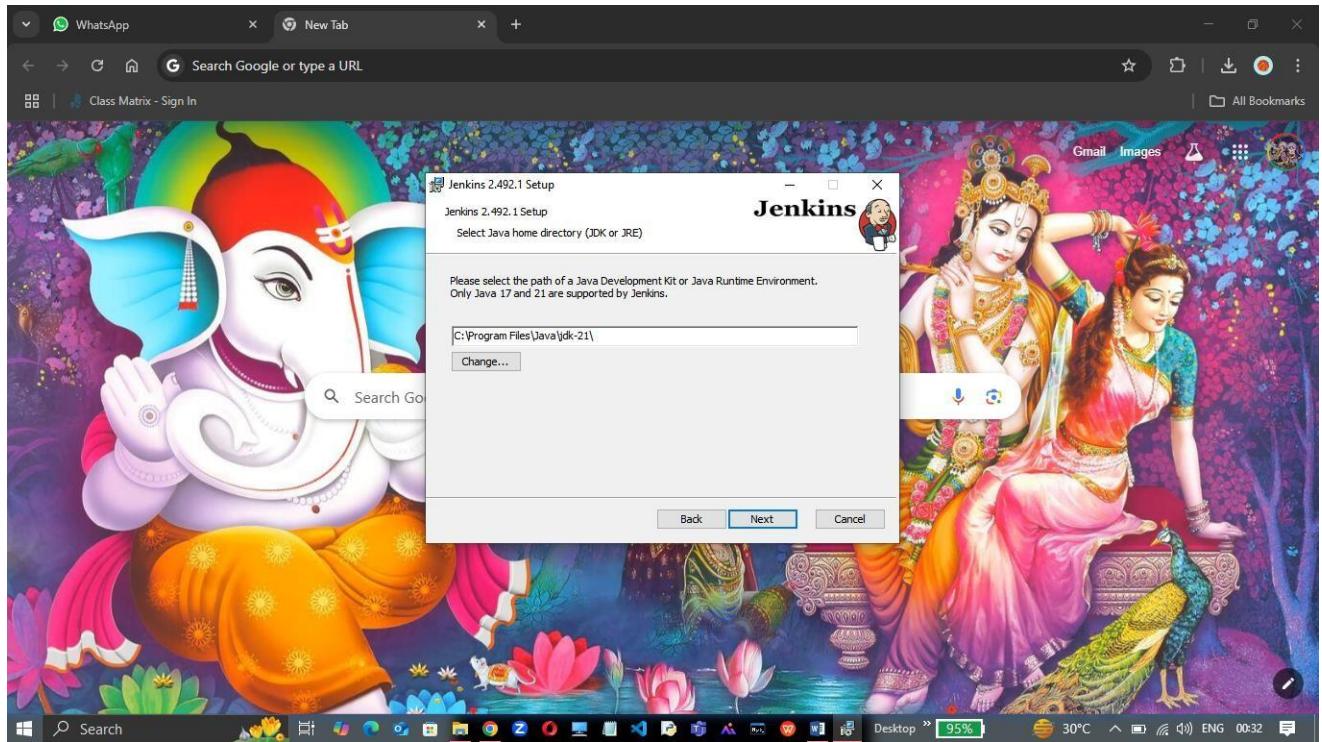
**Step 4:** Select the port 8080 and click Test Port button. The green tick will appear after which you can proceed to Next.



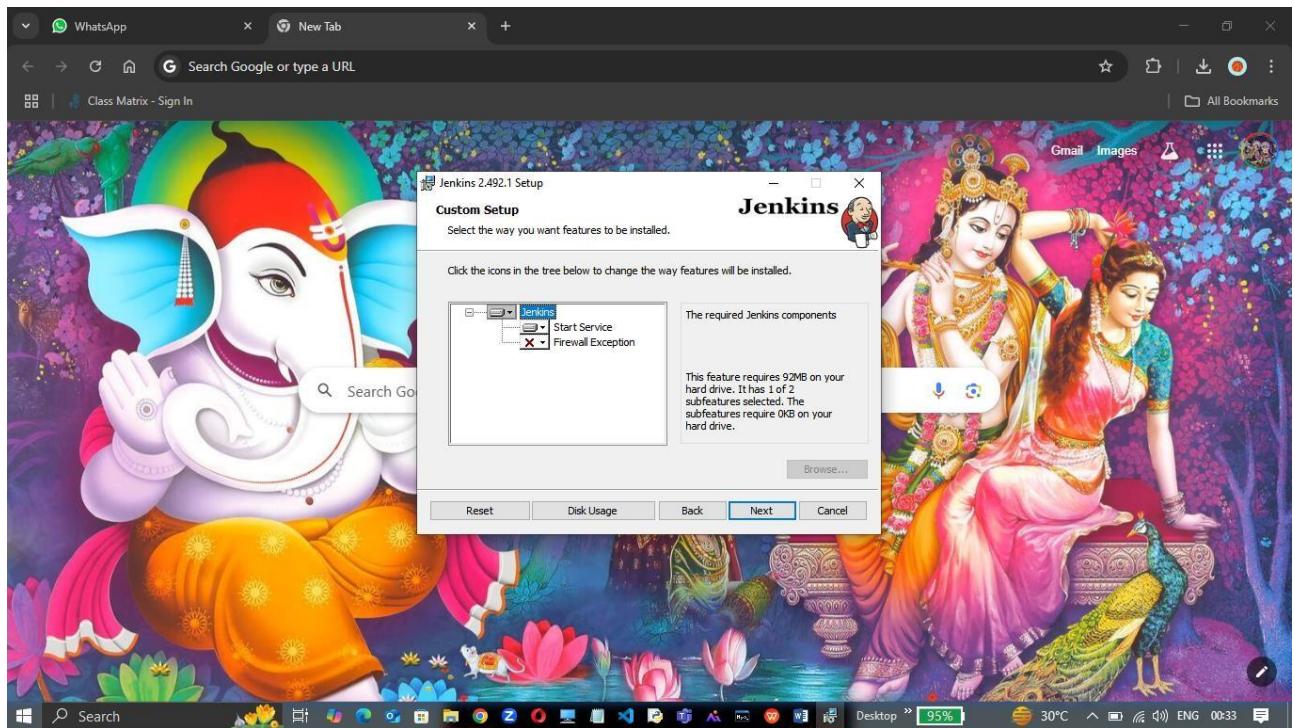
## Software Engineering & Project Management Lab Experiment No: - 04

**Aim:** To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job

**Step 5:** Locate the folder where you have installed JDK in the location path:

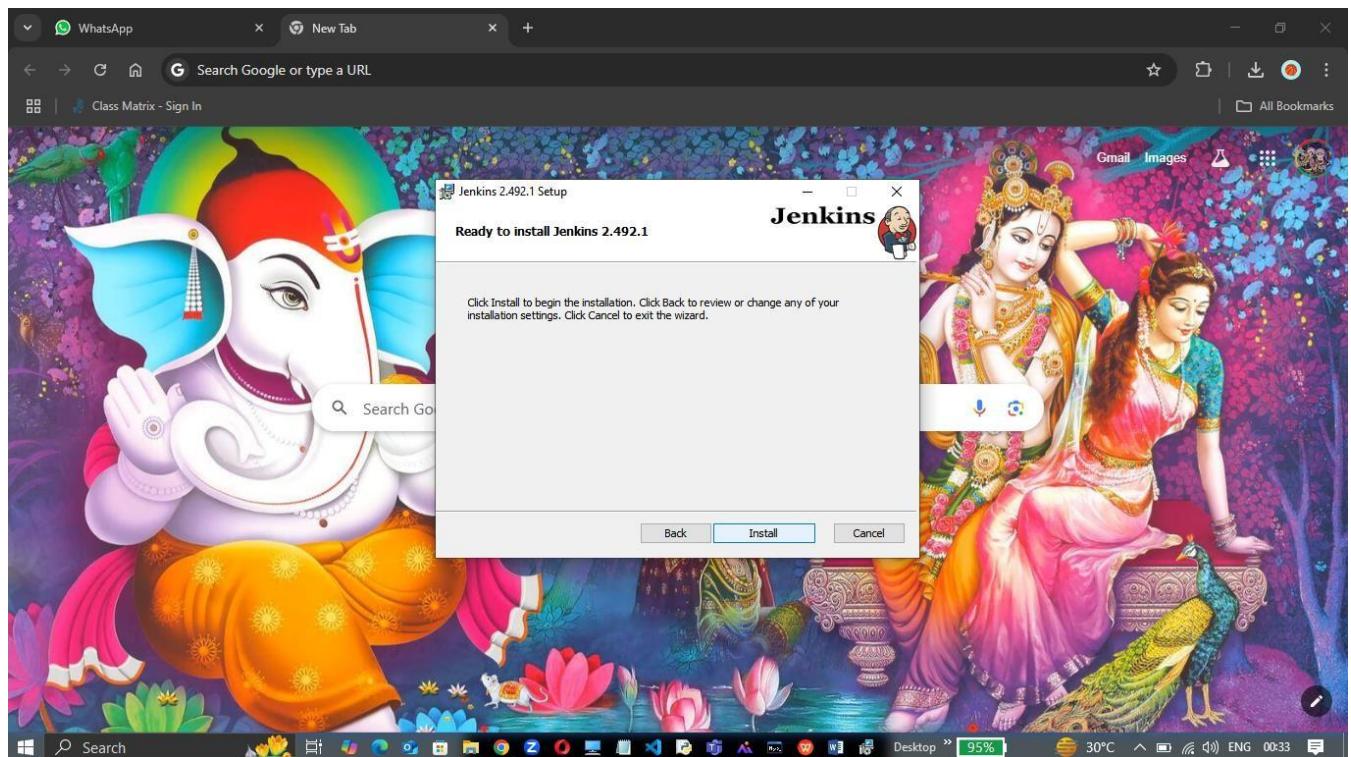


**Proceed to Next**

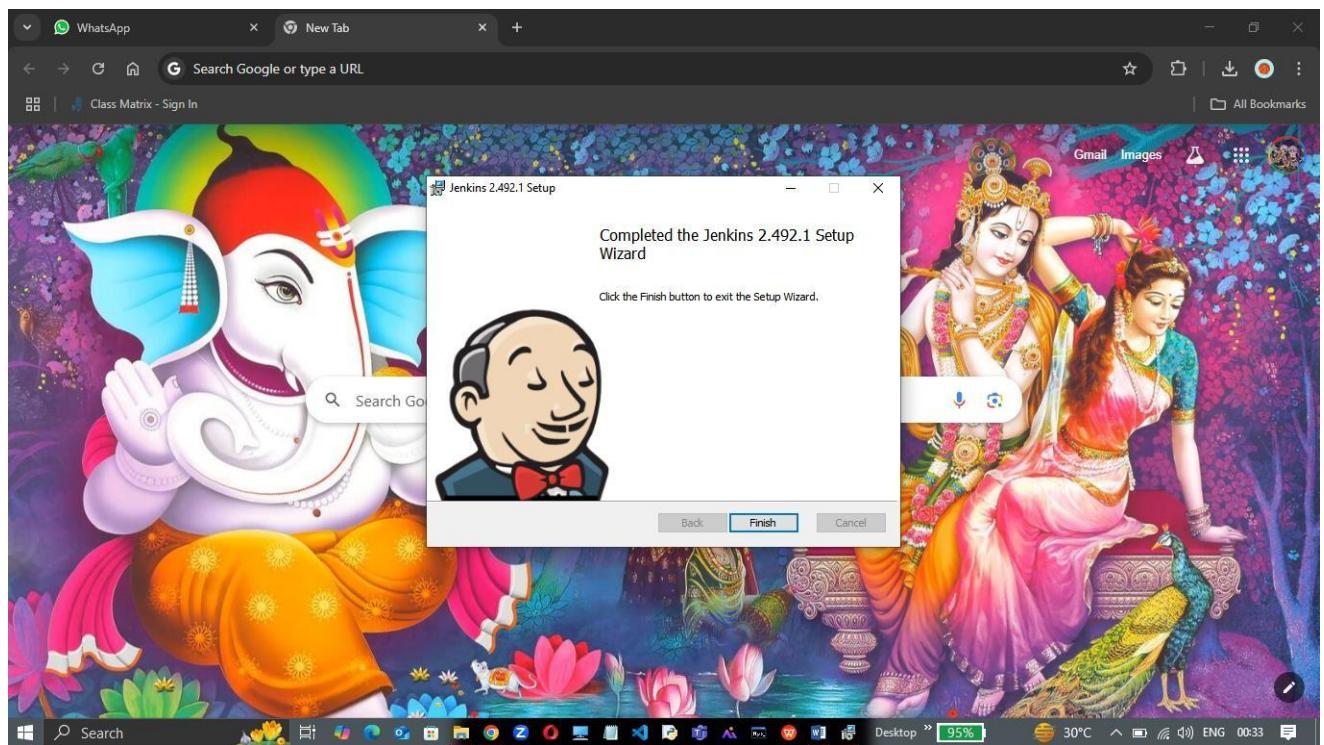


## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**



**On clicking 'Install', installation is finished.**

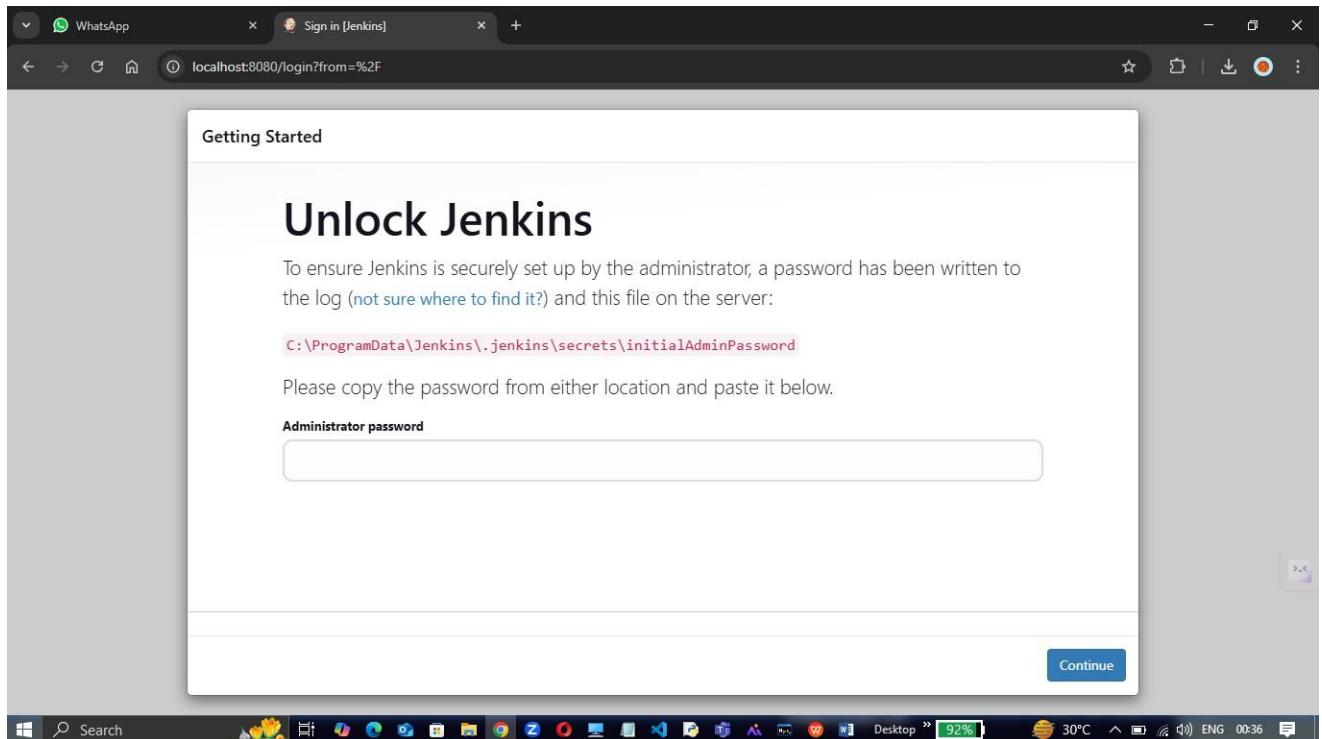


## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

**Step 6:** Once Installation is done, you can test the Jenkins on <http://localhost:8080> on the browser.

First time, when you open Jenkins portal it will ask to put admin default password which is stored in `/var/lib/jenkins/secrets/initialAdminPassword` file.



**Step 7:** On entering the password, you can continue to choose “Install Suggested Plugins”

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

Getting Started ×

# Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

## Install suggested plugins

Install plugins the Jenkins community finds most useful.

## Select plugins to install

Select and install plugins most suitable for your needs.

---

Jenkins 2.426.3

---

Once plugins are installed, click on next and specify the admin details along with the new password for Jenkins admin and click on finish to complete the installation.

After filling the details, click on Save & Continue, you will be redirected to the dashboard.

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

### Getting Started

# Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding	** bouncycastle API ** Instance Identity ** JavaBeans Activation Framework (JAF) API ** JavaMail API ** Credentials ** Plain Credentials ** Gson API ** Trilead API ** SSH Credentials Credentials Binding ** SCM API ** Pipeline: API ** commons-lang3 v3.x Jenkins API Timestamper ** Caffeine API ** Script Security ** JAXB ** SnakeYAML API ** Jackson 2 API ** commons-text API ** Pipeline: Supporting APIs ** Plugin Utilities API ** Font Awesome API ** Bootstrap 5 API ** JQuery3 API ** - required dependency
✓ Timestamper	⌚ Workspace Cleanup	⌚ Ant	⌚ Gradle	
⌚ Pipeline	⌚ GitHub Branch Source	⌚ Pipeline: GitHub Groovy Libraries	⌚ Pipeline: Stage View	
⌚ Git	⌚ SSH Build Agents	⌚ Matrix Authorization Strategy	⌚ PAM Authentication	
⌚ LDAP	⌚ Email Extension	⌚ Mailer		

Jenkins 2.426.3

Dashboard >

+ New Item      Add description

People      Build History      Manage Jenkins      My Views

**Welcome to Jenkins!**

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job      +

Set up a distributed build

Set up an agent      ↗  
Configure a cloud      ↕  
Learn more about distributed builds      ⓘ

Build Queue      No builds in the queue.

Build Executor Status      1 Idle      2 Idle

REST API      Jenkins 2.426.3

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

Getting Started

# Create First Admin User

Username

Password

Confirm password

Full name

E-mail address

Jenkins 2.426.3

[Skip and continue as admin](#) [Save and Continue](#)

Dashboard >

Enter an item name

example 1 » 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.

**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.

**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**  
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**  
Creates a set of multibranch project subfolders by scanning for repositories.

**OK**

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

The screenshot shows the Jenkins configuration interface for a job named 'example 1'. On the left, there's a sidebar with options like General, Source Code Management, Build Triggers, Build Environment (which is selected), Build Steps, and Post-build Actions. The main area is titled 'Build Steps' and contains a single step: 'Execute Windows batch command'. The command entered is 'echo "hello tsec"'. There are also checkboxes for adding timestamps to the console output, inspecting build logs, terminating stuck builds, and using Ant.

The screenshot shows the Jenkins console output for build #11 of the 'example 1' job. The left sidebar has links for Status, Changes, Console Output (which is selected), View as plain text, Edit Build Information, Delete build '#11', and Previous Build. The main content area is titled 'Console Output' and displays the following log:

```
Started by user Muskan Tolani
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\example 1
[example 1] $ cmd /c call C:\Windows\TEMP\jenkins6203665954710491391.bat
C:\ProgramData\Jenkins\.jenkins\workspace\example 1>echo "hello tsec"
"hello tsec"

C:\ProgramData\Jenkins\.jenkins\workspace\example 1>exit 0
Finished: SUCCESS
```

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

The screenshot shows the Jenkins dashboard with a list of projects. The table has columns for Status (S), Last Success (W), Name, Last Success, Last Failure, and Last Duration. Projects listed include example-1, My\_example, myex, and r.

S	W	Name	Last Success	Last Failure	Last Duration
Green	Sunny	example-1	1 yr 1 mo #1	N/A	0.17 sec
Red	Rainy	My_example	N/A	2 yr 0 mo #5	19 ms
Red	Cloudy	myex	2 yr 0 mo #3	2 yr 0 mo #8	0.72 sec
Red	Cloudy	r	2 yr 0 mo #3	1 yr 1 mo #4	0.71 sec

The screenshot shows the 'New Item' creation dialog. The user has entered 'TestProject' into the 'Enter an item name' field. Below it, a list of item types is shown:

- Freestyle project**: Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- 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.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**: Creates a set of multibranch project subfolders by scanning for repositories.

At the bottom is an 'OK' button.

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**

The screenshot shows the Jenkins configuration interface for a project named 'TestProject'. In the left sidebar, under 'Build Steps', the 'Execute shell' option is selected. A command box contains the text 'echo "Prasad"'. Below the command box is an 'Advanced' dropdown. At the bottom of the configuration page are 'Save' and 'Apply' buttons.

**Dashboard > TestProject > Configuration**

**Configure**

**Build Steps**

Automate your build process with ordered tasks like code compilation, testing, and deployment.

**Execute shell**

Command

See the list of available environment variables

```
echo "Prasad"
```

Advanced

Add build step

**Post-build Actions**

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Add post-build action

Save Apply

REST API Jenkins 2.492.1

The screenshot shows the Jenkins dashboard for the 'TestProject'. The 'Status' bar indicates the project is green. The 'Builds' section shows a single build labeled '#1 1:11PM' from today. On the right, there are links for 'Edit description', 'admin', and 'log out'.

**Jenkins**

Dashboard > TestProject >

**Status**

**TestProject**

This is a test project

**Workspace**

**Build Now**

**Configure**

**Delete Project**

**Rename**

**Builds**

Today #1 1:11PM

Edit description admin log out

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim:** To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job

The screenshot shows the Jenkins interface for a build job named 'TestProject'. The 'Console Output' tab is selected. The output shows the build was started by user 'admin' and ran as 'SYSTEM'. It displays the command 'Building in workspace C:\ProgramData\Jenkins\workspace\TestProject [TestProject] \$ C:\Windows\system32\cmd.exe -xe C:\WINDOWS\TEMP\jenkins10724806366536973722.sh' and the Microsoft Windows version 'Microsoft Windows [Version 10.0.22631.4896]'. The copyright notice '(c) Microsoft Corporation. All rights reserved.' is also present. The build completed successfully with the message 'C:\ProgramData\Jenkins\workspace\TestProject>Finished: SUCCESS'.

The screenshot shows a terminal window with a dark background. On the right, there are two collapsed sections labeled 'Code' and 'bash'. The terminal output shows the creation of a bash script 'example1.sh' with the following content:

```
15L@203-009 MINGW64 ~
$ cat > example1.sh
#!/bin/bash
name=$1
Address=$2
echo "Hello $name ..your address is $Address"
```

After saving the file, the terminal shows the command 'cat > example1.sh' followed by '[1]+ Stopped'. The prompt then changes to '15L@203-009 MINGW64 ~' and shows the command '\$'.

The terminal window continues with the following session:

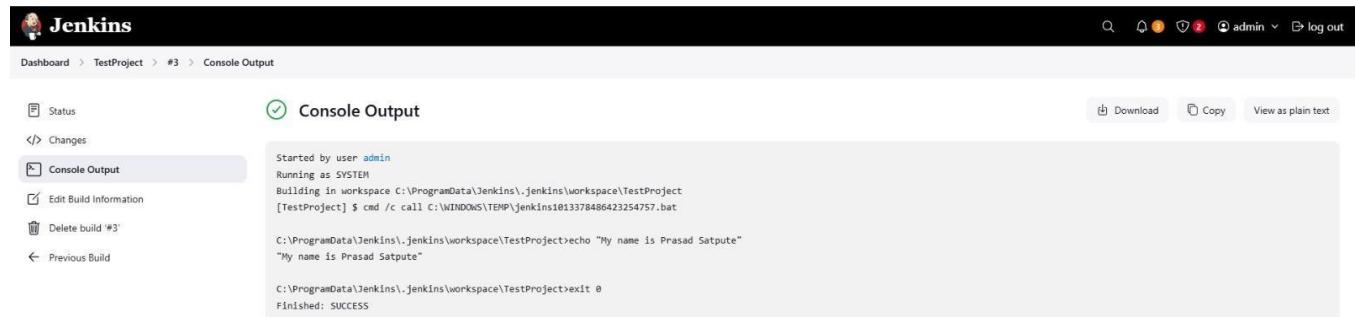
```
15L@203-009 MINGW64 ~
$ bash example1.sh
Hello ..your address is

15L@203-009 MINGW64 ~
$ bash example1.sh Prasad
Hello Prasad ..your address is

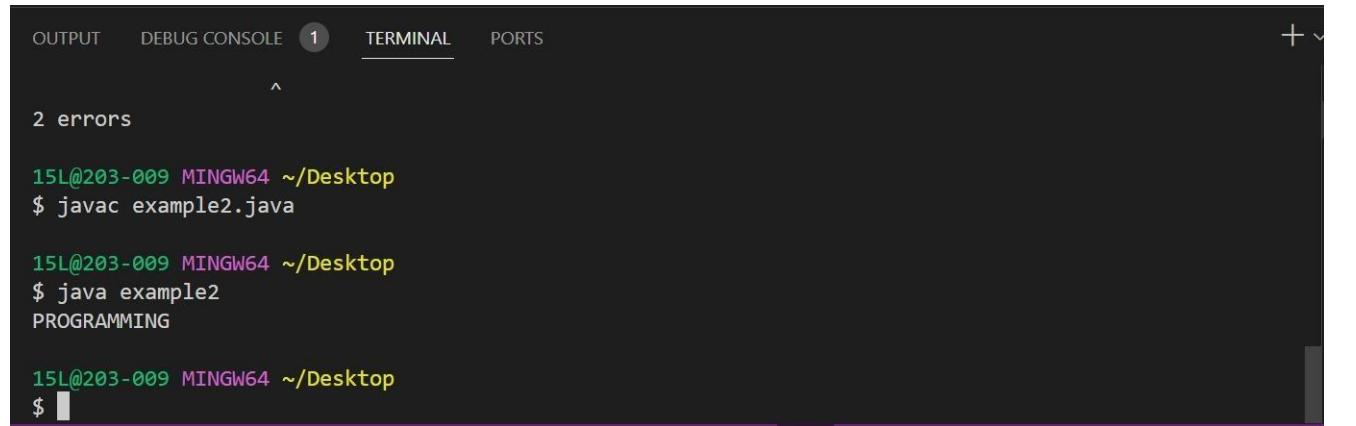
15L@203-009 MINGW64 ~
$ bash example1.sh Prasad Santacruz
Hello Prasad ..your address is Santacruz
```

## Software Engineering & Project Management Lab Experiment No: - 04

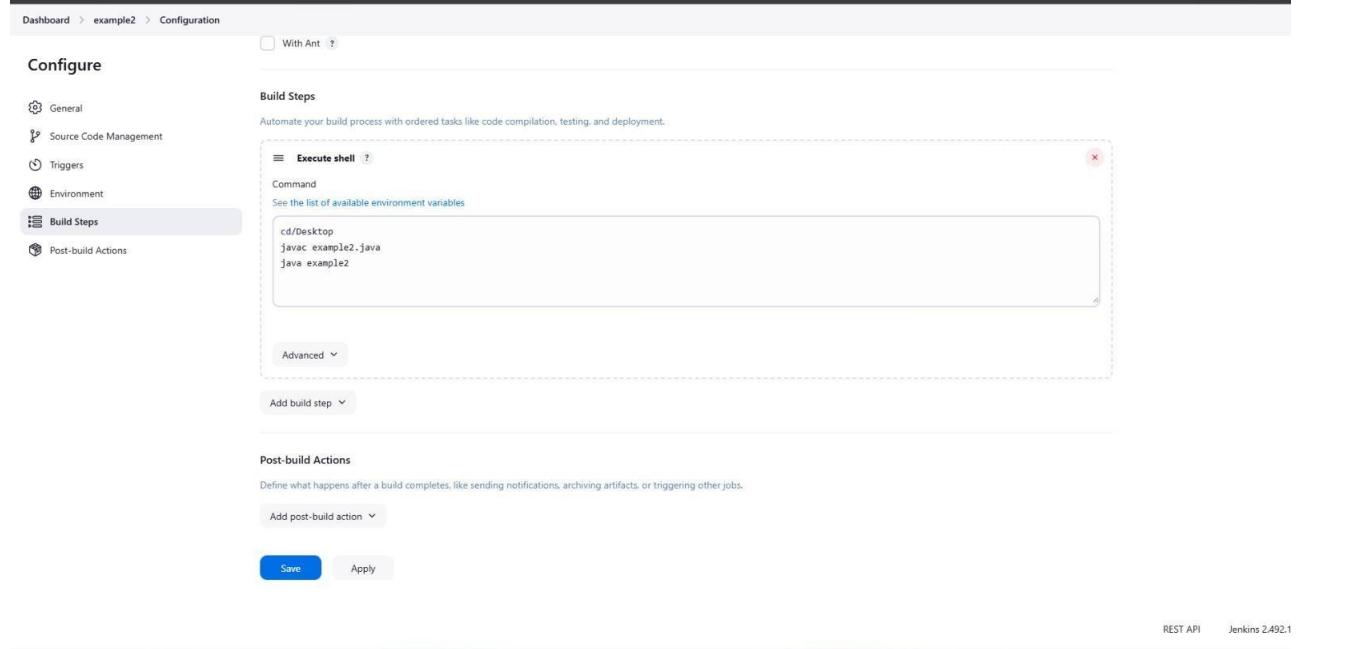
**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**



The screenshot shows the Jenkins console output for a build named 'TestProject' #3. The log starts with 'Started by user admin' and 'Running as SYSTEM'. It shows the command being run: 'cmd /c call C:\WINDOWS\TEMP\jenkins1013378486423254757.bat'. The output then displays the echo command: 'C:\ProgramData\Jenkins\jenkins\workspace\TestProject>echo "My name is Prasad Satpute"' followed by 'My name is Prasad Satpute'. Finally, it shows the exit command: 'C:\ProgramData\Jenkins\jenkins\workspace\TestProject>exit 0' and 'Finished: SUCCESS'.



The screenshot shows a terminal window with three lines of output. The first line shows the terminal prompt '15L@203-009 MINGW64 ~/Desktop' and the command '\$ javac example2.java'. The second line shows the terminal prompt '15L@203-009 MINGW64 ~/Desktop' and the command '\$ java example2'. The third line shows the output of the Java program, which is 'PROGRAMMING'.



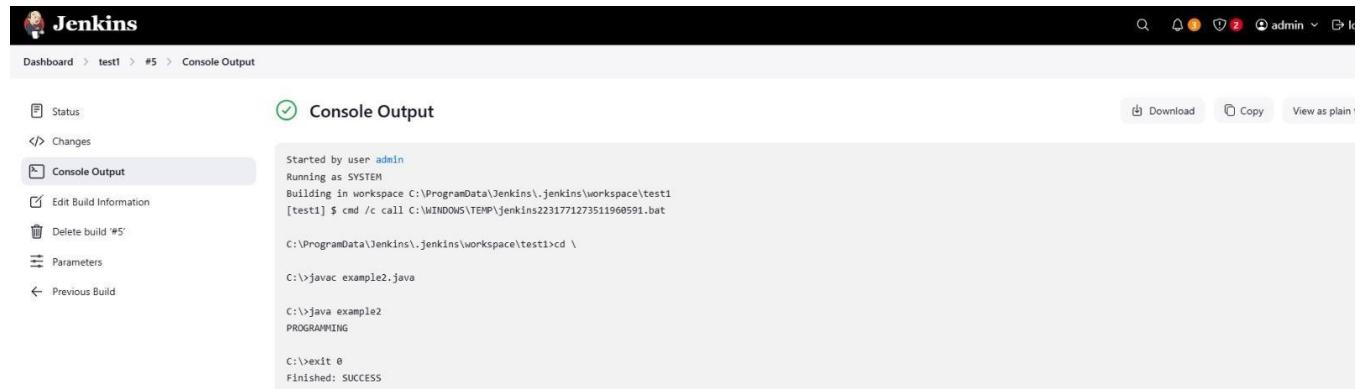
The screenshot shows the Jenkins configuration page for the 'example2' job. Under the 'Build Steps' section, there is an 'Execute shell' step with the following command entered:

```
cd/Desktop  
javac example2.java  
java example2
```

Below the build steps, there is a 'Post-build Actions' section with a placeholder for adding actions.

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**



The screenshot shows the Jenkins interface for the 'test1' job. The 'Console Output' tab is selected. The output window displays the following command-line session:

```
Started by user admin
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\test1
[test1] $ cmd /c call C:\Windows\TEMP\jenkins2231771273511960591.bat

C:\ProgramData\Jenkins\jenkins\workspace\test1>cd \
C:\>javac example2.java

C:\>java example2
PROGRAMMING

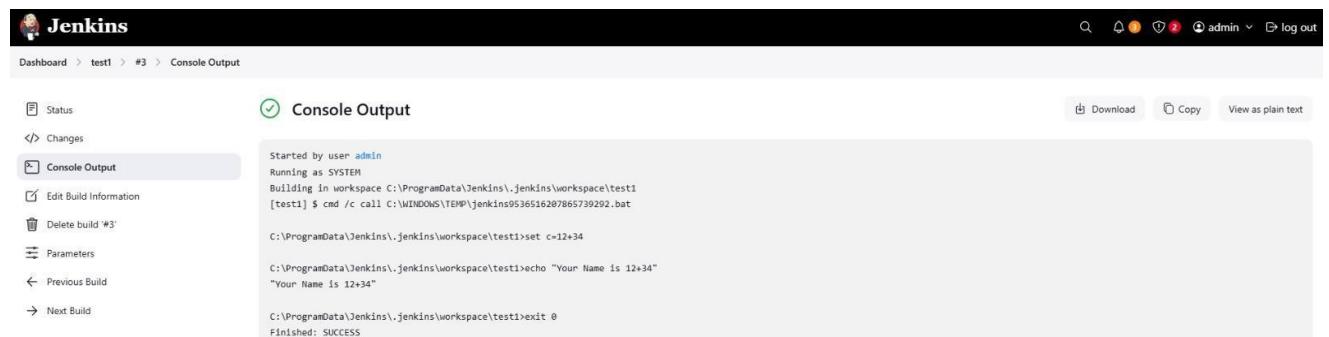
C:\>exit 0
Finished: SUCCESS
```



The screenshot shows the Jenkins interface for the 'test1' job. The 'Console Output' tab is selected. The output window displays the following command-line session:

```
Started by user admin
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\test1
[test1] $ cmd /c call C:\Windows\TEMP\jenkins149301988206271570.bat

C:\ProgramData\Jenkins\jenkins\workspace\test1>set /a c=+2
C:\ProgramData\Jenkins\jenkins\workspace\test1>echo "Your Name is 3"
"Your Name is 3"
C:\ProgramData\Jenkins\jenkins\workspace\test1>exit 0
Finished: SUCCESS
```



The screenshot shows the Jenkins interface for the 'test1' job. The 'Console Output' tab is selected. The output window displays the following command-line session:

```
Started by user admin
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\test1
[test1] $ cmd /c call C:\Windows\TEMP\jenkins9536516287865739292.bat

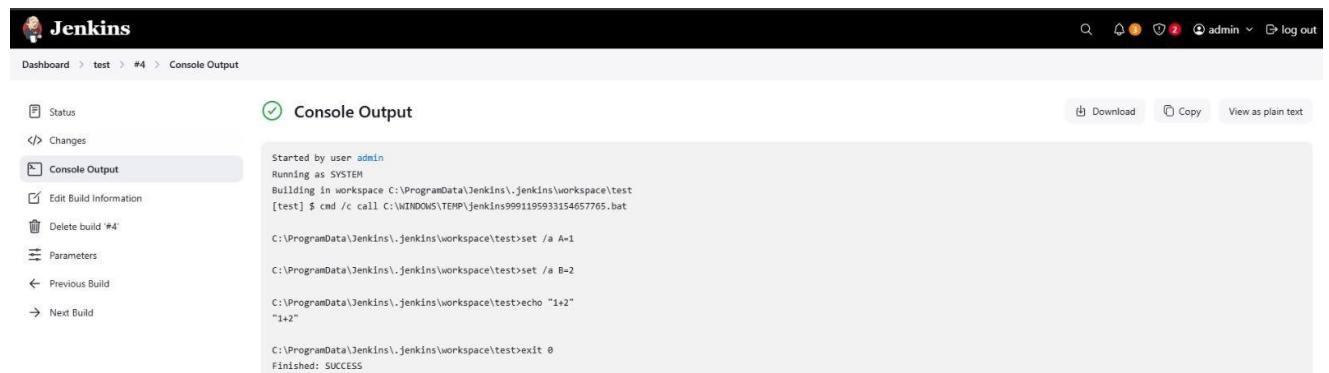
C:\ProgramData\Jenkins\jenkins\workspace\test1>set c=12+34
C:\ProgramData\Jenkins\jenkins\workspace\test1>echo "Your Name is 12+34"
"Your Name is 12+34"
C:\ProgramData\Jenkins\jenkins\workspace\test1>exit 0
Finished: SUCCESS
```

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**



Started by user admin  
Running as SYSTEM  
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\test1  
[test1] \$ cmd /c call C:\WINDOWS\TEMP\jenkins3591631450106967559.bat  
C:\ProgramData\Jenkins\jenkins\workspace\test1>echo "Your Name is Sachin"  
"Your Name is Sachin"  
C:\ProgramData\Jenkins\jenkins\workspace\test1>exit 0  
Finished: SUCCESS



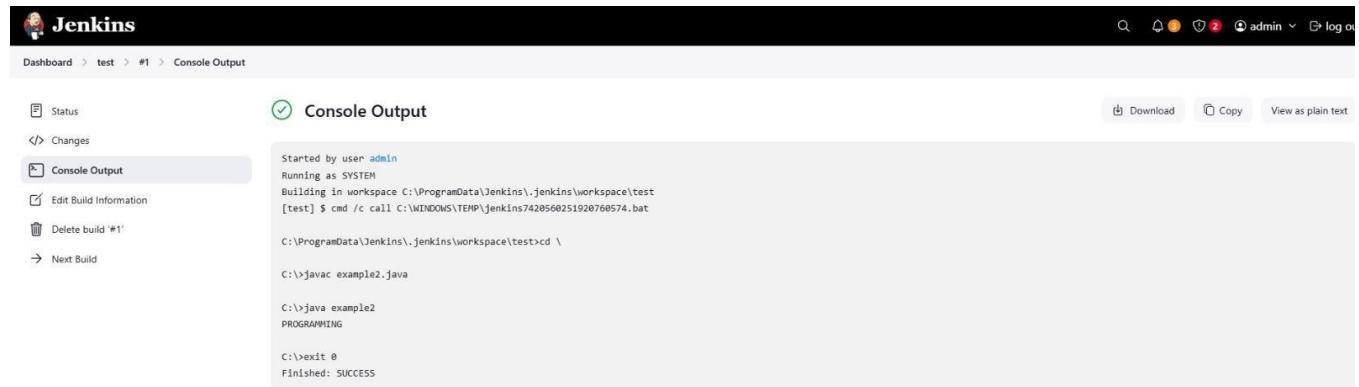
Started by user admin  
Running as SYSTEM  
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\test  
[test] \$ cmd /c call C:\WINDOWS\TEMP\jenkins991195933154657765.bat  
C:\ProgramData\Jenkins\jenkins\workspace\test>set /a A=1  
C:\ProgramData\Jenkins\jenkins\workspace\test>set /a B=2  
C:\ProgramData\Jenkins\jenkins\workspace\test>echo "1+2"  
"1+2"  
C:\ProgramData\Jenkins\jenkins\workspace\test>exit 0  
Finished: SUCCESS



Started by user admin  
Running as SYSTEM  
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\test  
[test] \$ cmd /c call C:\WINDOWS\TEMP\jenkins2360247137534955462.bat  
C:\ProgramData\Jenkins\jenkins\workspace\test>echo "ABC and DEF"  
"ABC and DEF"  
C:\ProgramData\Jenkins\jenkins\workspace\test>exit 0  
Finished: SUCCESS

## Software Engineering & Project Management Lab Experiment No: - 04

**Aim: To understand Continuous Integration, install and configure Jenkins with Maven/Ant/Gradle to setup a build Job**



The screenshot shows the Jenkins interface for a build named 'test' under job '#1'. The 'Console Output' tab is selected. The output window displays the following command-line session:

```
Started by user admin
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\test
[test] $ cmd /c call C:\Windows\TEMP\jenkins7420560251920760574.bat

C:\ProgramData\Jenkins\jenkins\workspace\test>cd \
C:\>javac example2.java

C:\>java example2
PROGRAMMING

C:\>exit 0
Finished: SUCCESS
```

**Conclusion:** Thus, we have successfully installed and configured Jenkins with Maven/Ant/Gradle to setup a build Job and learnt about the implementation of Jenkins in open source continuous integration.

Name : Aarya Thakur

T22

Roll No: 114

## Experiment No-5

Aim: Experiment 5:To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins, create a pipeline script to Test and deploy an application over the tomcat server

Programming in Jenkins:

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible." In simple way, Continuous integration (CI) is the practice of frequently building and testing each change done to your code automatically.

Jenkins is a self-contained, open-source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

Our first job will execute the shell commands. The freestyle project provides enough options and features to build the complex jobs that you will need in your projects.

Example 1

Example 1.1: Deploying a freestyle app in Jenkins

Creating a job:

**Start building your software project**

Create a job

+

Naming the job and setting it as freestyle:

## Enter an item name

Example1

» Required field



### Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



### 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.



### Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.



### Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.



### Multibranch Pipeline

Creates a set of Pipeline projects according to detected branches in one SCM repository.



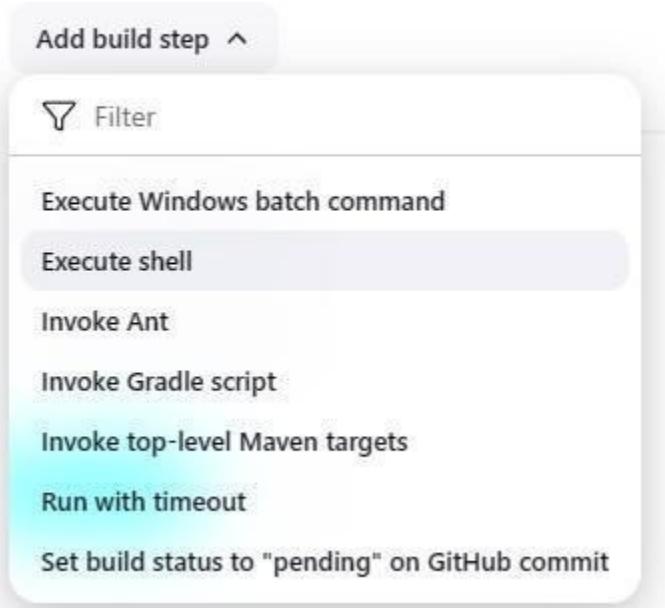
### Organization Folder

Creates a set of multibranch project subfolders by scanning for repositories.

OK

Selecting build type as “Execute shell”:

## Build Steps



Entering a simple command for the shell execution:



Applying and saving the project configuration:

Save

Apply

Saved

Building the project:

Build Now

Console output (after building):

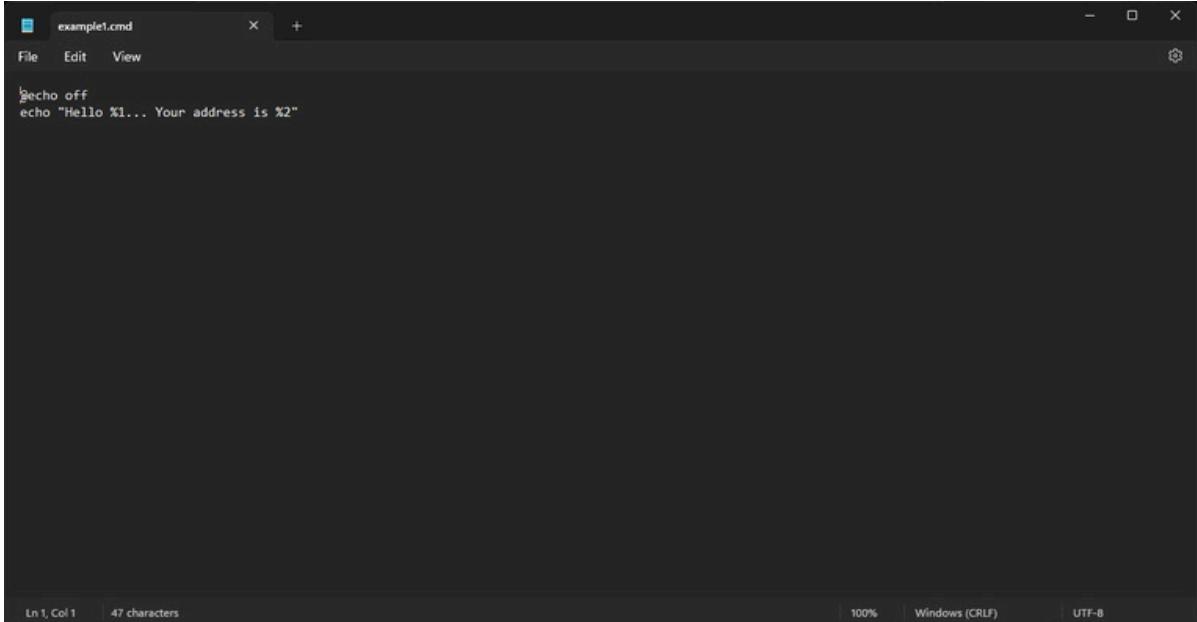


The screenshot shows the Jenkins 'Console Output' page. It displays the build log for a job named 'Example'. The log shows the script being run, its execution, and a successful completion message.

```
Started by user Siddhant Chetlur
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\workspace\Example1
[Example1] $ "C:\Program Files\Git\bin\sh.exe" -xe C:\WINDOWS\TEMP\jenkins15583852534385450955.sh
+ echo 'Hello TSEC'
Hello TSEC
Finished: SUCCESS
```

### Example 1.2: Taking parameters through files

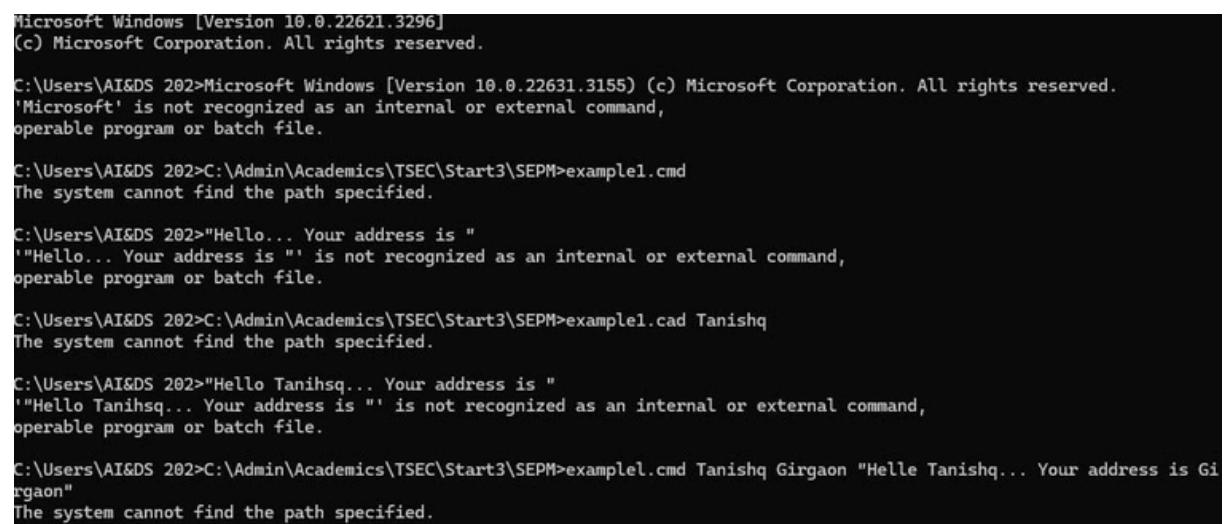
Contents of script example1.cmd:



The screenshot shows a Windows Notepad window titled 'example1.cmd'. It contains a simple batch script that uses the 'echo' command to print a greeting and a placeholder for an address.

```
echo off
echo "Hello %1... Your address is %2"
```

Executing script example1.cmd on the terminal:



The screenshot shows a terminal window on Windows. The user attempts to execute the 'example1.cmd' script from the command line. The terminal shows several failed attempts due to incorrect paths and command recognition.

```
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\AI&DS 202>Microsoft Windows [Version 10.0.22631.3155] (c) Microsoft Corporation. All rights reserved.
'Microsoft' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPM>example1.cmd
The system cannot find the path specified.

C:\Users\AI&DS 202>"Hello... Your address is "
'"Hello... Your address is "' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPM>example1.cad Tanishq
The system cannot find the path specified.

C:\Users\AI&DS 202>"Hello Tanishq... Your address is "
'"Hello Tanishq... Your address is "' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPM>example1.cmd Tanishq Girgaon "Hello Tanishq... Your address is Girgaon"
The system cannot find the path specified.
```

Modifying the Jenkins project to execute the script while supplying required parameters:

## Build Steps

The screenshot shows the Jenkins build step configuration for an 'Execute Windows batch command' step. The step is defined with the following command:

```
C:\Admin\Academics\TSEC\Start3\SEPM\example1.cmd Siddhant Goregaon
```

Below the command, there is an 'Advanced' dropdown menu and a 'Add build step' button.

Console output after building the modified project:

The screenshot shows the Jenkins console output page for the build. The 'Console Output' tab is selected. The output log shows the following:

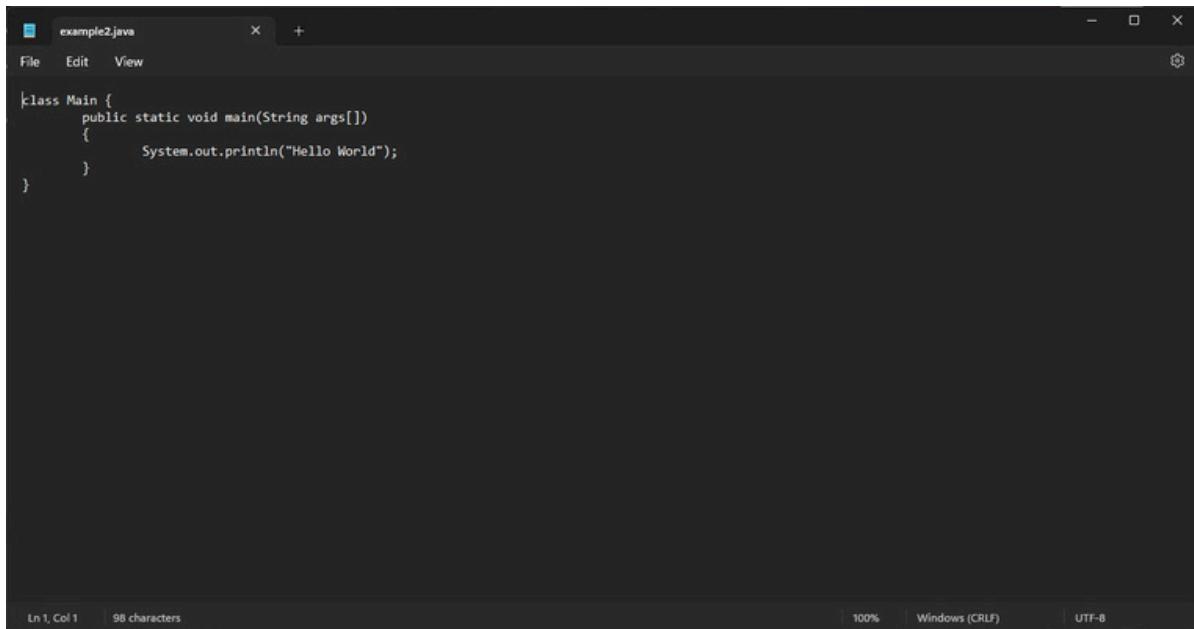
```
Started by user Siddhant Chettur
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example1
[example1] $ cmd /c call C:\Windows\TEMP\jenkins70750958165161158.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example1\cmd Hello Siddhant... Your address is Goregaon
"Hello Siddhant... Your address is Goregaon"
Finished: SUCCESS
```

## Example 2

### Example 2.1: Running a Java program under Jenkins

Creating a simple Java program:

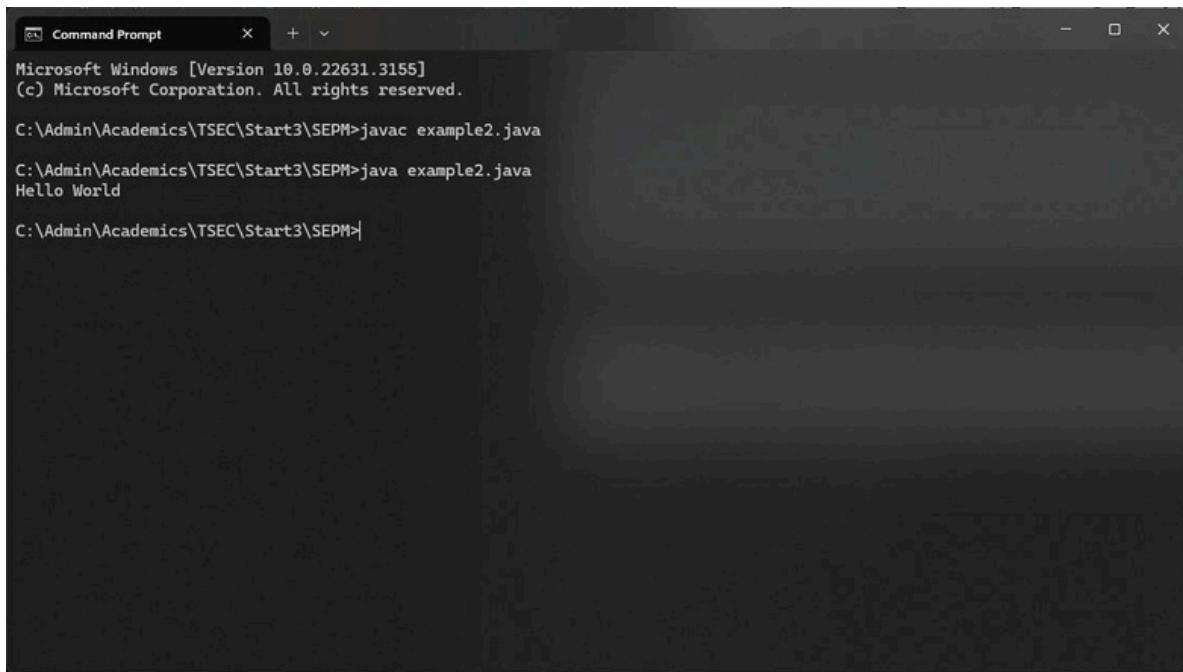


```
example2.java
```

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

In 1, Col 1    98 characters    100%    Windows (CRLF)    UTF-8

Compiling and running the program on the terminal:

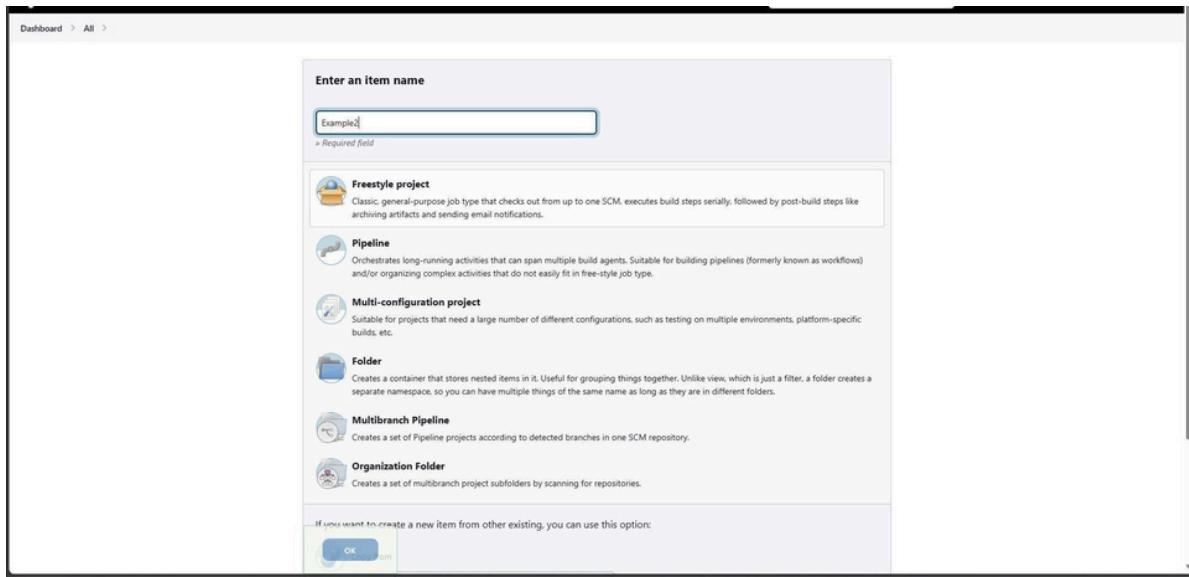


```
Command Prompt
```

```
Microsoft Windows [Version 10.0.22631.3155]
(c) Microsoft Corporation. All rights reserved.

C:\Admin\Academics\TSEC\Start3\SEPM>javac example2.java
C:\Admin\Academics\TSEC\Start3\SEPM>java example2.java
Hello World
C:\Admin\Academics\TSEC\Start3\SEPM>
```

## Creating a new freestyle project:



## Configure new project:

The screenshot shows the 'Build Steps' configuration screen. It features a single step: 'Execute Windows batch command'. The 'Command' field contains the following script:  
javac C:\Admin\Academics\TSEC\Start3\SEPM\example2.java  
java C:\Admin\Academics\TSEC\Start3\SEPM\example2.java

## Console output after building:

The screenshot shows the Jenkins 'Console Output' log. It displays the following text:  
Started by user Siddhant Chetlur  
Running as SYSTEM  
[EnvInject] - Loading node environment variables.  
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example2  
[Example2] \$ cmd /c call C:\WINDOWS\TEMP\jenkins15296462484398614135.bat  
C:\ProgramData\Jenkins\jenkins\workspace\Example2>javac C:\Admin\Academics\TSEC\Start3\SEPM\example2.java  
C:\ProgramData\Jenkins\jenkins\workspace\Example2>java C:\Admin\Academics\TSEC\Start3\SEPM\example2.java  
Hello World  
C:\ProgramData\Jenkins\jenkins\workspace\Example2>exit 0  
Finished: SUCCESS

## Example 3

### Example 3.1: Parameterise build

Creating a new freestyle project:

**Enter an item name**

Example3  
» Required field

**Freestyle project**  
 Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**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.

**Multi-configuration project**  
 Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**  
 Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

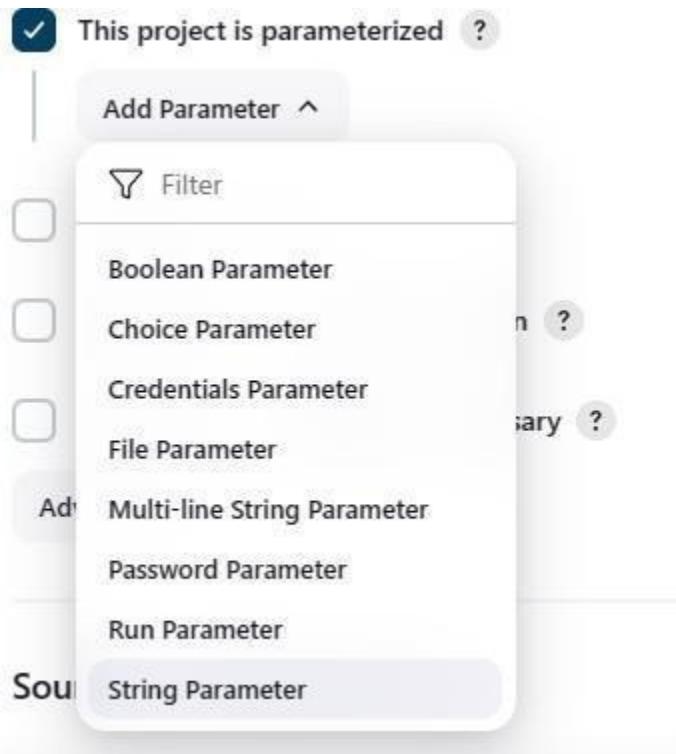
**Multibranch Pipeline**  
 Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**  
 Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

**OK**

Enabling parameterisation and adding a String parameter:



Configuring the string parameter as Fname :

The screenshot shows the configuration dialog for a 'String Parameter'. The title bar says 'String Parameter'. The form fields are as follows:

- Name**: Fname
- Default Value**: (empty)
- Description**: (empty)
- Plain text**: Preview
- Trim the string

Adding a choice parameter and configuring it as `City` with the following choices:

The screenshot shows a configuration dialog for a "Choice Parameter". The "Name" field is set to "City". The "Choices" field contains a list of six items: Bandra, Kalyan, Dombivali, Churchgate, Thane, and Dadar. The "Description" field is empty. At the bottom, there are "Plain text" and "Preview" buttons.

Creating a script which takes 2 arguments for name and city:

```
C:\Users\AI&DS 202>Microsoft Windows [Version 10.0.22631.3155] (c) Microsoft Corporation. All rights reserved.  
'Microsoft' is not recognized as an internal or external command,  
operable program or batch file.  
  
C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPH>example3.cmd  
The system cannot find the path specified.  
  
C:\Users\AI&DS 202>Hello your name is and your city is  
'Hello' is not recognized as an internal or external command,  
operable program or batch file.  
  
C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPH example3.cmd Tanishq  
The system cannot find the path specified.  
  
C:\Users\AI&DS 202>Hello your name is Tanishq and your city is  
'Hello' is not recognized as an internal or external command,  
operable program or batch file.  
  
C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPM>example3.cmd Tansishq Bandra  
The system cannot find the path specified.  
  
C:\Users\AI&DS 202>Hello your name is Tanishq and your city is Bandra  
'Hello' is not recognized as an internal or external command,  
operable program or batch file.  
  
C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPH
```

Configuring build steps:

The screenshot shows a "Build Steps" configuration. It includes an "Execute Windows batch command" step. The "Command" field contains the command: `C:\Admin\Academics\TSEC\Start3\SEPM\example3.cmd %Fname% %City%`. An "Advanced" dropdown menu is open at the bottom left, showing options like "Add build step".

Entering parameters for build:

## Project Example3

This build requires parameters:

Fname

City

Build

Cancel

Console output after building:

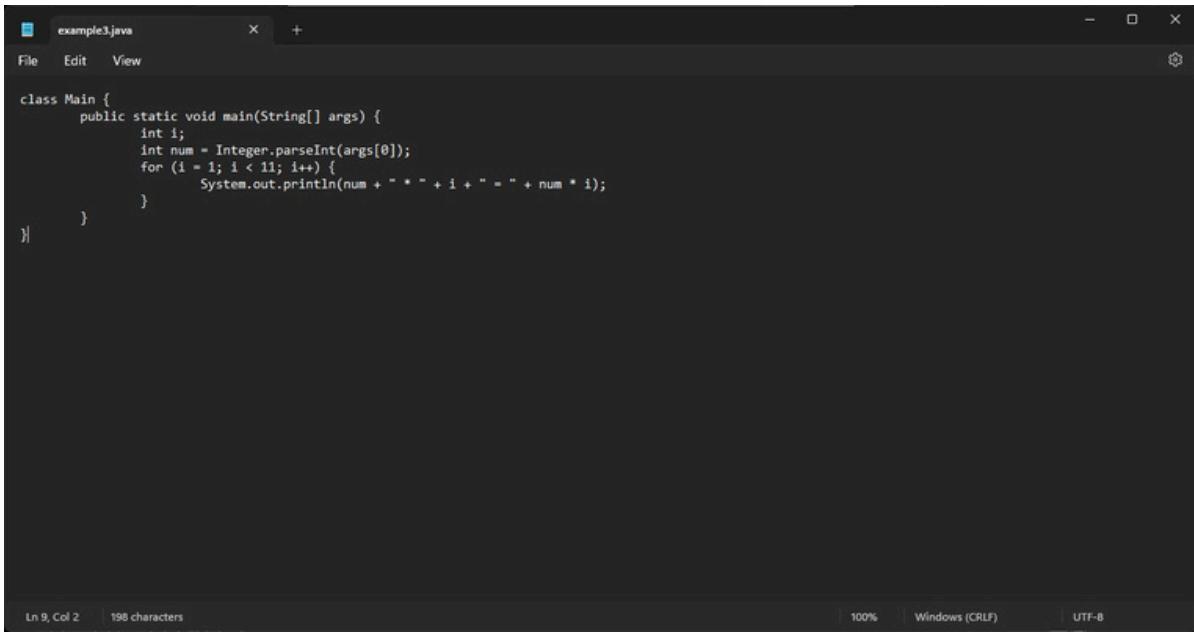
### Console Output

```
Started by user Siddhant Chetlur
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example3
[Example3] $ cmd /c call C:\WINDOWS\TEMP\jenkins14094536165150986151.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example3>C:\admin\Academics\TSEC\Start3\SEPM\example3.cmd Siddhant Bandra
Hello your name is Siddhant and your city is Bandra
Finished: SUCCESS
```

## Example 3.2: Running a Java program with parameters

Creating a Java program with an input argument:



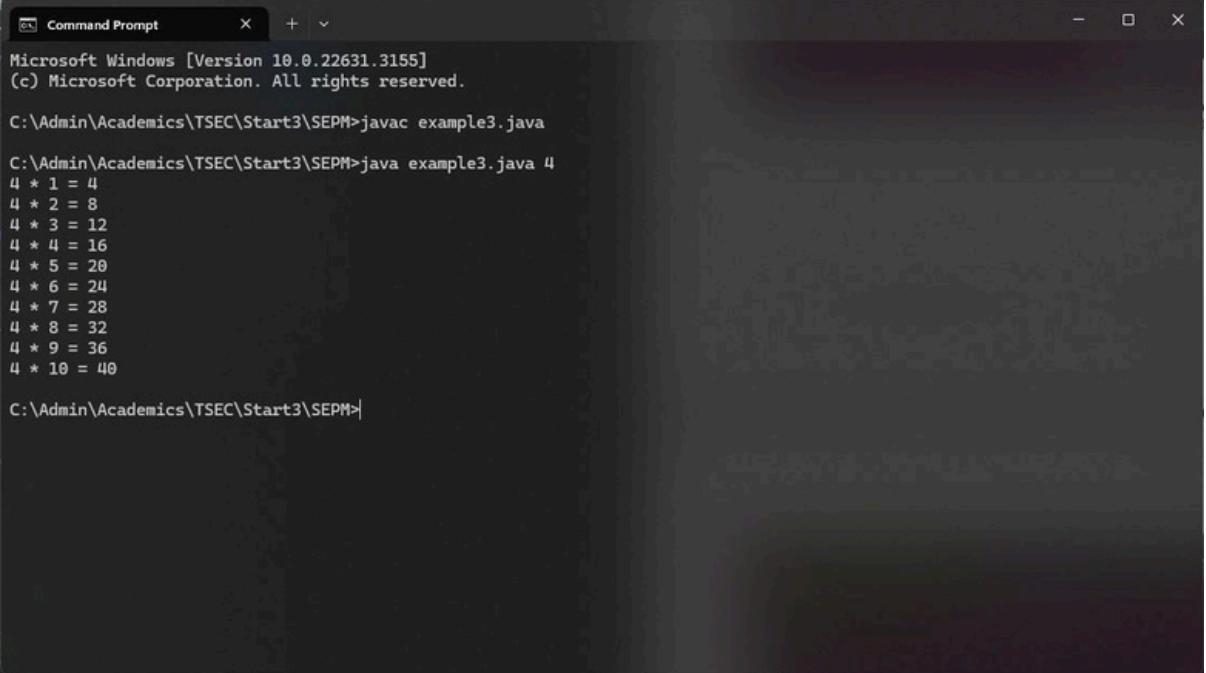
```
example3.java

File Edit View

class Main {
    public static void main(String[] args) {
        int i;
        int num = Integer.parseInt(args[0]);
        for (i = 1; i < 11; i++) {
            System.out.println(num + " * " + i + " = " + num * i);
        }
    }
}

Ln 9, Col 2 198 characters 100% Windows (CRLF) UTF-8
```

Testing the program on the terminal:

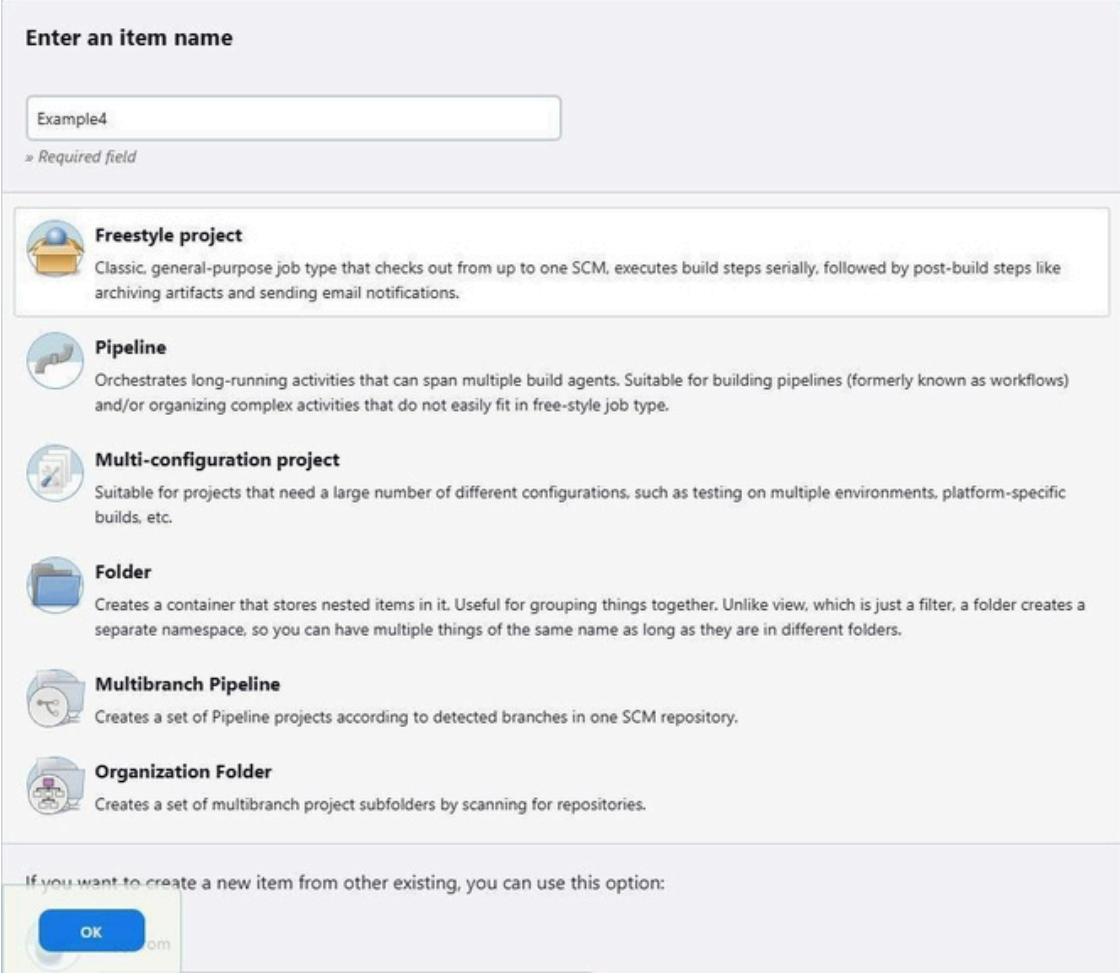


```
Microsoft Windows [Version 10.0.22631.3155]
(c) Microsoft Corporation. All rights reserved.

C:\Admin\Academics\TSEC\Start3\SEPM>javac example3.java
C:\Admin\Academics\TSEC\Start3\SEPM>java example3.java 4
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
4 * 10 = 40

C:\Admin\Academics\TSEC\Start3\SEPM>
```

Creating a new freestyle project:



Enter an item name

Example4  
» Required field

**Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**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.

**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**  
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**  
Creates a set of multibranch project subfolders by scanning for repositories.

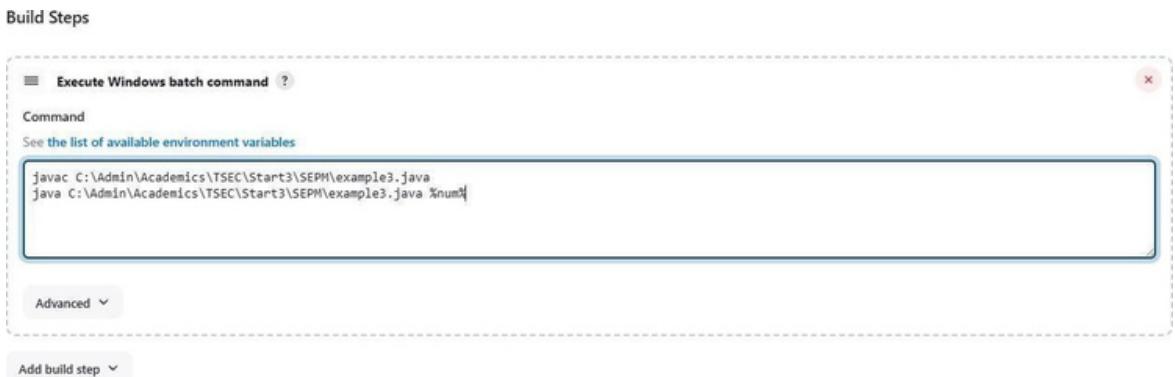
If you want to create a new item from other existing, you can use this option:

OK

Parameterise the project by adding a string parameter as follows:



Configure the build steps:



Entering the parameter for the build:

#### Project Example4

This build requires parameters:

num

25

Build

Cancel

Console output after building:

 **Console Output**

```
Started by user Siddhant Chetlur
Running as SYSTEM
[EnvInject] - Loading node environment variables...
Building in workspace C:\ProgramData\Jenkins\workspace\Example4
[Example4] $ cmd /c call C:\WINDOWS\TEMP\jenkins15119185770823247700.bat

C:\ProgramData\Jenkins\workspace\Example4>javac C:\Admin\Academics\TSEC\Start3\SEPM\example3.java

C:\ProgramData\Jenkins\workspace\Example4>java C:\Admin\Academics\TSEC\Start3\SEPM\example3 25
25 * 1 = 25
25 * 2 = 50
25 * 3 = 75
25 * 4 = 100
25 * 5 = 125
25 * 6 = 150
25 * 7 = 175
25 * 8 = 200
25 * 9 = 225
25 * 10 = 250

C:\ProgramData\Jenkins\workspace\Example4>exit 0
Finished: SUCCESS
```

## Example 5

### Example 5.1: Running a Python program

Creating a simple Python script:

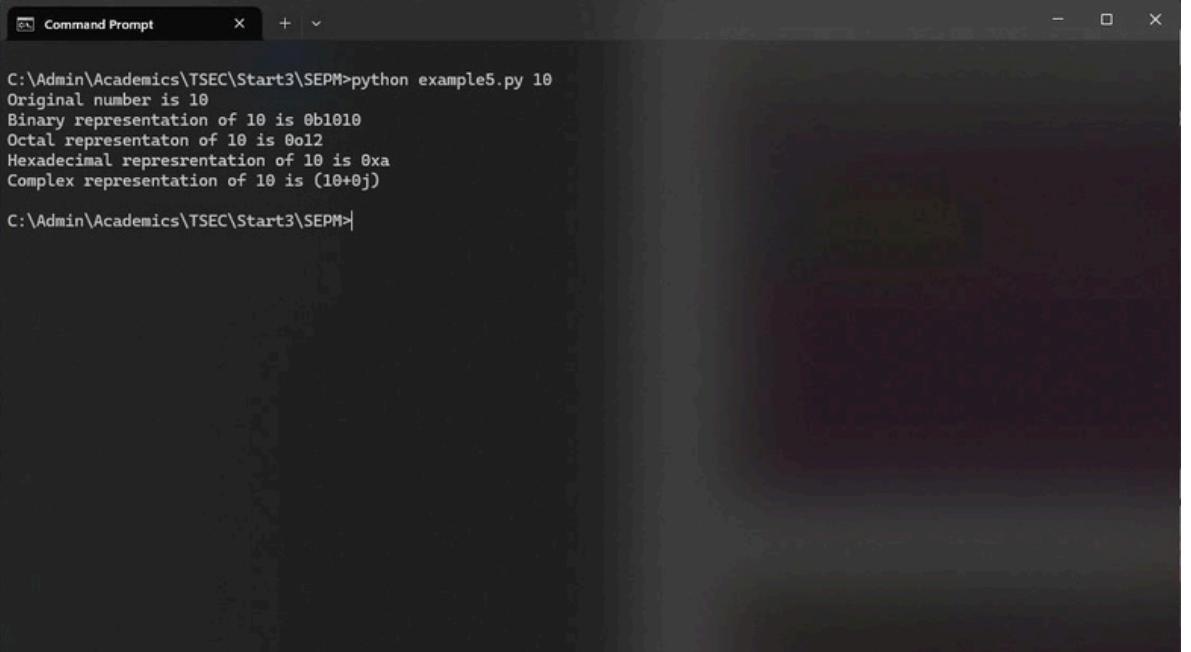


```
example5.py

import sys
num = int(sys.argv[1])
print(f"Original number is {num}")
print(f"Binary representation of {num} is {bin(num)}")
print(f"Octal representaton of {num} is {oct(num)}")
print(f"Hexadecimal representation of {num} is {hex(num)}")
print(f"Complex representation of {num} is {complex(num)}")
```

In 9, Col 60    299 characters    100%    Windows (CRLF)    UTF-8

Running the Python script on the terminal:



```
C:\Admin\Academics\TSEC\Start3\SEPM>python example5.py 10
Original number is 10
Binary representation of 10 is 0b1010
Octal representaton of 10 is 0o12
Hexadecimal representation of 10 is 0xa
Complex representation of 10 is (10+0j)

C:\Admin\Academics\TSEC\Start3\SEPM>
```

Creating a new freestyle project:

**Enter an item name**

Example5  
» Required field

 **Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

 **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.

 **Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

 **Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

 **Multibranch Pipeline**  
Creates a set of Pipeline projects according to detected branches in one SCM repository.

 **Organization Folder**  
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:  
**OK**

Parameterising the project with a string parameter as follows:

This project is parameterized ?

**String Parameter** ? x

Name ?  
num

Default Value ?

Description ?

Plain text [Preview](#)

Trim the string ?

Add Parameter ▾

## Configuring the build steps:

Build Steps

Execute Windows batch command [?](#)

Command

See [the list of available environment variables](#)

```
python C:\Admin\Academics\TSEC\Start3\SEPM\example5.py %num%
```

Advanced [▼](#)

Add build step [▼](#)

## Setting the parameter for the build:

### Project Example5

This build requires parameters:

num

10

[▷ Build](#)

[Cancel](#)

## Console output after building:

### Console Output

```
Started by user Siddhant Chetlur
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example5
[Example5] $ cmd /c call C:\WINDOWS\TEMP\jenkins11157306491994478222.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example5>python C:\Admin\Academics\TSEC\Start3\SEPM\example5.py 10
Original number is 10
Binary representation of 10 is 0b1010
Octal representation of 10 is 0o12
Hexadecimal representation of 10 is 0xa
Complex representation of 10 is (10+0j)

C:\ProgramData\Jenkins\jenkins\workspace\Example5>exit 0
Finished: SUCCESS
```

Conclusion: Thus, we have successfully studied Continuous Integration and installed, configured, and understood programming with Jenkins.



## EXPERIMENT 6

### STUDYING AGILE METHODOLOGY AND TEST CASE MANAGEMENT USING JIRA TOOL

Theory:

#### 1. Introduction to Agile Methodology

Agile methodology is an iterative approach to software development and project management that emphasizes flexibility, collaboration, and customer feedback. It helps teams deliver high-quality software in incremental steps rather than following a rigid plan.

##### 1.1 Agile Manifesto

Agile is based on four core values:

1. Individuals and interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Customer collaboration over contract negotiation.
4. Responding to change over following a plan.

##### 1.2 Agile Principles

Agile follows 12 principles, including:

- Customer satisfaction through early and continuous delivery.
- Embracing changing requirements for the customer's competitive advantage.
- Delivering working software frequently.
- Close collaboration between business and development teams.
- Motivated individuals and self-organizing teams.
- Continuous attention to technical excellence.

#### 2. Agile Frameworks

Several frameworks exist under Agile, including:

1. Scrum – A framework for iterative and incremental product development.
2. Kanban – A visual workflow management system.
3. Extreme Programming (XP) – Focuses on engineering practices.

4. Lean Software Development – Focuses on minimizing waste and maximizing value.

## 2.1 Scrum Framework

Scrum is one of the most popular Agile frameworks. It consists of:

- Scrum Team: Product Owner, Scrum Master, Development Team.
- Scrum Artifacts: Product Backlog, Sprint Backlog, Increment.

- Scrum Events: Sprint Planning, Daily Stand-ups, Sprint Review, Sprint Retrospective.
- Sprint Cycle: A time-boxed period (usually 2-4 weeks) where a team completes selected backlog items.

## 3. Introduction to Jira

Jira is a popular tool developed by Atlassian for Agile project management, issue tracking, and test case management. It helps teams plan, track, and manage software development projects efficiently.

### 3.1 Features of Jira

- Customizable dashboards for tracking project progress.
- Issue tracking for creating, assigning, and monitoring tasks.
- Workflow automation to improve efficiency.
- Integration with various development and testing tools.

## 4. Test Case Management Using Jira

Jira can be used to manage test cases effectively by integrating with testing plugins like Zephyr and Xray.

### 4.1 Steps to Manage Test Cases in Jira

#### 1. Creating a Test Case:

- o Navigate to the Jira project.
- o Click on Create Issue and select Test Case (if using Zephyr/Xray).
- o Define the test case with details such as Test Summary, Preconditions, Test Steps, and Expected Results.
- o Click Create to save the test case.

#### 2. Executing a Test Case:

- o Open the created test case.
- o Click Execute and select the test cycle.
- o Update the test execution status (Pass, Fail, In Progress, Blocked).

#### 3. Tracking Test Results:

- o Use Jira dashboards to generate reports on test execution progress.
- o Analyze defects by linking test cases to bug reports.

#### 4. Integrating Jira with CI/CD Pipelines:

- o Jira can integrate with Jenkins, Bamboo, and other DevOps tools to automate testing workflows.

## 5. Benefits of Using Jira for Agile and Test Case Management

- Centralized Tracking: All tasks, sprints, and test cases are managed in one place.
- Collaboration: Teams can communicate and update task statuses in real-time.

- Customization: Jira can be tailored to fit any workflow.
- Automation: Reduces manual effort through workflow automation and integration with DevOps tools.

Output:

The screenshot shows the Jira homepage with the following elements:

- Header:** Jira logo, navigation links (Features, Product guide, Templates, Pricing, Enterprise), "Get it free" button, search bar, and "Sign in" link.
- Main Callout:** "Great outcomes start with Jira" with a wavy underline.
- Text:** "The only project management tool you need to plan and track work across every team."
- Sign-up Form:** "Work email" input field containing "thakur.aarya23@gmail.com", a note about separating work and life emails, and a "Sign up" button.
- Or continue with:** Buttons for Google and Microsoft.
- Product Cards:**
  - Software Development:** Product & Issue Tracking, "IN PROGRESS" status, "Optimize experience for mobile web" card, "FEEDBACK" button, and "NUC-335" issue card.
  - Marketing:** Plan & Launch Campaigns, "Marketing" section, calendar view for "SUN" and "MON" (27, 28), and a "FIN-23 Workshop with..." card.
  - IT:** Plan & Track IT Projects, "IT" section, "All changes saved" note, "Licensing and billing form" card, and a "FIN-23 Workshop with..." card.
  - Design:** Build Creative Workflows, "Design" section, "EXP-21 / EXP-1" integration, "Implement integrations into" dropdown, "Attach" and "Create subtask" buttons, and a "Designs (1)" card for "Figma".
  - Operations:** Create Custom Processes, "Operations" section, "Assignee" and "Due" columns for "Alana Song" (17 Aug), "Fran Perez" (29 Sep), and "Andres Ramos" (07 Nov), and a "Privacy + Terms" link.

Atlassian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Atlassian cookies and tracking notice](#)

Preferences Only necessary ✓ Accept all

Jira Your

My Scrum Project (Software project)

PLANNING

- Summary
- Timeline
- Backlog**
- Board
- Forms
- + Add view

DEVELOPMENT

- Code
- Project pages
- Project settings

Archived issues NEW

+ Create issue

Welcome!

Your first project is ready to kick off. It's where you'll track tasks across teams, turning big ideas into real outcomes.

Name your project

e.g. My Scrum Project

How familiar are you with Jira?\*

- Not familiar
- Somewhat
- Familiar

Get started

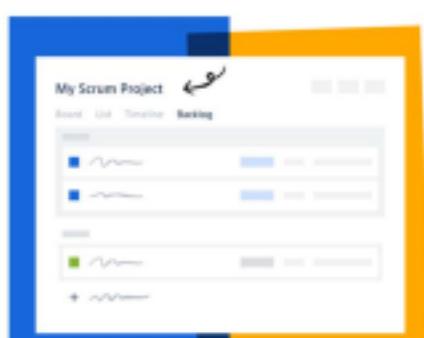
My Scrum Project

Board List Timeline Backlog

Start sprint

Create sprint

Quickstart



Atlassian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Atlassian cookies and tracking notice](#)

Preferences Only necessary ✓ Accept all

Jira Your

My Scrum Project (Software project)

PLANNING

- Summary
- Timeline
- Backlog**
- Board
- Forms
- + Add view

DEVELOPMENT

- Code
- Project pages
- Project settings

Archived issues NEW

+ Create issue

Welcome!

Your first project is ready to kick off. It's where you'll track tasks across teams, turning big ideas into real outcomes.

Name your project

SEPMLab6

How familiar are you with Jira?\*

- Not familiar
- Somewhat
- Familiar

Get started

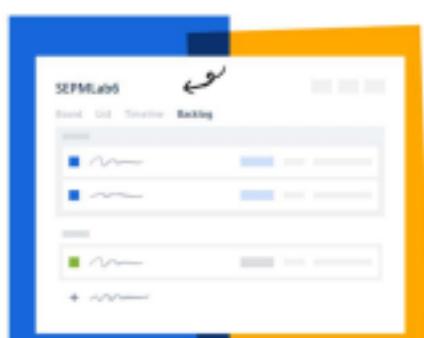
SEPMLab6

Board List Timeline Backlog

Start sprint

Create sprint

Quickstart



SEPMLab6 Software project

Projects / My Scrum Project

## Backlog

Search  vw B Epic Insights View settings

SCRUM Sprint 1 Add dates 0 issues Start sprint ...

Plan your sprint  
Drag issues from the Backlog section, or create new issues, to plan the work for this sprint.  
Select Start sprint when you're ready.

+ Create issue

Backlog 0 issues Create sprint

Your backlog is empty.

+ Create issue

Dismiss Quickstart

This screenshot shows the Jira Backlog page for the 'My Scrum Project'. On the left, a sidebar lists project management sections like Planning, Development, and Jira Core. The main area displays 'SCRUM Sprint 1' with 0 issues and a 'Start sprint' button. Below it is the 'Backlog' section, which is currently empty. A 'Create issue' button is available. A 'Quickstart' sidebar on the right provides links to create a project, an issue, and a sprint, along with a video tutorial and basic setup steps.

Jira Your work Projects Filters Dashboards Teams Plans Apps Create 30 days left Search

SEPMLab6 Software project

Timeline

Search timeline Give feedback Share Export ...

Sprints

		FEB
SCRUM-1 Web Development	DONE	
SCRUM-2 Flutter		
SCRUM-3 API_Gangster		
SCRUM-4 Data Science	DONE	

+ Create Epic

Today Weeks Months Quarters

Dismiss Quickstart

This screenshot shows the Jira Timeline page for the 'SEPMLab6' project. The sidebar includes standard Jira navigation. The main timeline shows four sprints: 'SCRUM-1 Web Development' (DONE), 'SCRUM-2 Flutter', 'SCRUM-3 API\_Gangster', and 'SCRUM-4 Data Science' (DONE). The timeline is set to show 'FEB'. A 'Quickstart' sidebar on the right is identical to the one in the Backlog view, providing initial setup steps and a video tutorial.



## Select a template to get started

You can always change this later. Selecting a template won't limit what you can do.

### Other

**Scrum**  
Plan, prioritize, and schedule sprints using scrum framework

**Kanban**  
Visualize work and maximize efficiency with a kanban board

**Project management**  
Manage, track, and assign tasks to streamline project workflows

Back

Continue

## Conclusion

This experiment provided an overview of Agile methodology and how Jira is used for project and test case management. Jira enhances Agile practices by improving workflow efficiency, collaboration, and test case execution tracking, making it a valuable tool in modern software development.

## Experiment 7

**Aim:** To Create Scrum Board for Scrum Master using JIRA Tool.

### Theory:

To create a Scrum Board for a Scrum Master using the JIRA tool, you need to follow these steps:

Create a Scrum Project in JIRA:

Log in to your JIRA account and select a template from the library, choosing the Scrum template.

Once the project is created, you will land on the empty backlog, also known as the product backlog, containing a list of potential work items for the project

Create User Stories or Tasks in the Backlog:

In JIRA, work items like user stories, tasks, and bugs are referred to as "issues." Create user stories using the quick create option on the backlog.

User stories describe work items in a non-technical language from a user's perspective, following the format: "As a {type of user}, I want {goal} so that I {receive benefit}"

Prioritize User Stories in the Backlog:

After creating user stories, prioritize them by ranking or dragging and dropping them in the order they should be worked on.

The product owner usually prioritizes user stories, and the development team estimates the effort required to complete each story

Create a Scrum Board in JIRA:

Click on Search > View all boards and then Create board.

Select the board type as Scrum and choose whether to start with a new project

template or add the board to existing projects.

Configure columns and quick filters to reflect your team's process and focus on specific issues quickly

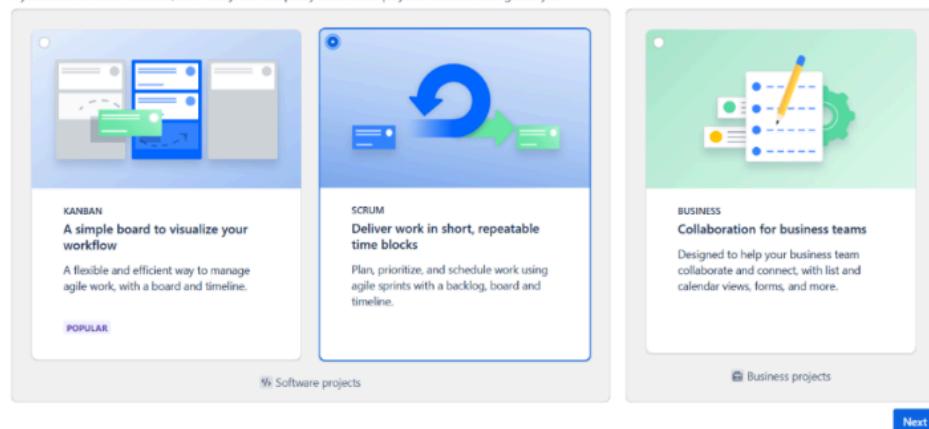
### Navigate Between Boards:

Use the board switcher located in the left-hand menu under the project name to move between different boards in JIRA

By following these steps, a Scrum Master can effectively create a Scrum Board in JIRA, enabling efficient management of tasks, user stories, and sprints within the agile project management framework.

#### Select a template for your first project

If you're not sure what to choose, don't worry. You can quickly create a new project if this one's not right for you.



The screenshot shows the Atlassian Admin interface. On the left, there's a sidebar with various administrative settings like Details, Domains, API keys, Application tunnels, Emails, Contacts, Compliance (marked as NEW), Platform experiences (marked as BETA), Data management, AI Settings, Rovo (marked as NEW), Atlassian Intelligence (marked as NEW), Migration to cloud, and Product links.

The main content area is titled "Platform experiences". It says: "Platform experiences are features that come with any plan and can be used across Atlassian products. Goals and projects are currently in early access and available to everyone on the sites below. You can opt out of early access at any time. [Get more info on early access](#)".

Site	Activated on	Actions
	Mar 25, 2025	<a href="#">Opt out</a>

At the bottom, there are navigation buttons for "Previous" and "Next" and a "Results per page" dropdown set to 5.

Atlassian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Atlassian cookies and tracking notice](#)

Preferences Only necessary ✓

# Welcome!

Your first project is ready to kick off. It's where you'll track tasks across teams, turning big ideas into real outcomes.

Name your project

How familiar are you with Jira?\*

Not familiar  
 Somewhat  
 Familiar

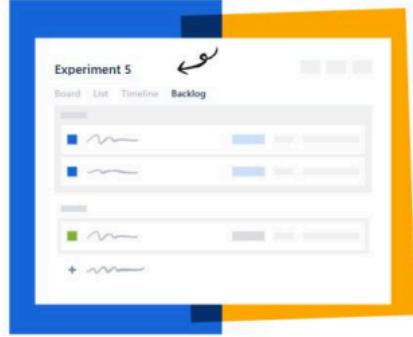
[Get started](#)

You're in a team-managed project

Learn more [+ Create issue](#)

Your backlog is empty.

[Create sprint](#) [Quickstart](#)



Atlassian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Atlassian cookies and tracking notice](#)

Preferences Only necessary ✓

# Welcome!

Your first project is ready to kick off. It's where you'll track tasks across teams, turning big ideas into real outcomes.

Name your project

How familiar are you with Jira?\*

Not familiar  
 Somewhat  
 Familiar

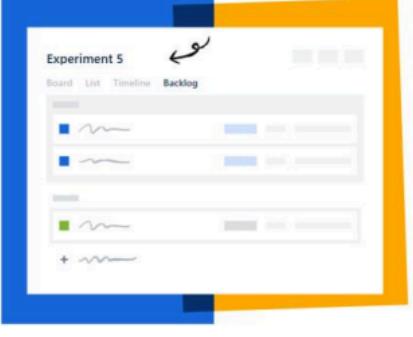
[Get started](#)

You're in a team-managed project

Learn more [+ Create issue](#)

Your backlog is empty.

[Create sprint](#) [Quickstart](#)



Atlassian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Atlassian cookies and tracking notice](#)

Preferences Only necessary ✓ Accept all

Jira Your work Projects Filters Dashboards Teams Plans Apps Create 10 days left Search

## Projects

Search Projects  Filter by product

Create project Templates

Name	Key	Type	Lead	Project URL	More actions
Experiment 5	SCRUM	Team-managed software			...

< 1 >

**Templates**  
Preview a template for your next project

- Scrum Deliver work in short time blocks
- Product roadmap TRY Create custom roadmaps
- Work requests TRY Manage unplanned work
- Top-level planning PREMIUM Monitor work across your teams

[More templates](#)

Atlassian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Atlassian cookies and tracking notice](#)

Preferences Only necessary ✓ Accept all

The screenshot shows a Jira interface for a project titled "SCRUM Sprint 1". A modal window titled "Added people to project" is open, stating "You have added 2 people to your project and 2 new licenses have been added to Jira." It lists two users: "aaryathakur48@gmail.com New to Jira" and "raj123@gmail.com New to Jira". There is a "Project settings" button and an "OK" button.

Atlassian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Atlassian cookies and tracking notice](#)

Preferences Only necessary ✓ Accept all

The screenshot shows a Jira interface for a project titled "SCRUM-1". A modal window titled "Choose a topic" is open, with the sub-section "Choose a topic" selected. It lists four items: "SCRUM-1", "SCRUM-2", "SCRUM-3", and "SCRUM-4", each with a "Edit" icon. Below the modal, the main Jira board shows a "TO DO 4" column with the same four items. On the right, a detailed view of "SCRUM-1" is shown, including fields for Assignee (Aarya Thakur), Labels (None), Parent (None), Team (None), Sprint (SCRUM Sprint 1), Story point estimate (None), and Reporter (None).

Atlassian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Atlassian cookies and tracking notice](#)

Preferences Only necessary ✓ Accept all

The screenshot shows a Jira interface for a project titled "SCRUM". A modal window titled "Add Flag" is open, with the sub-section "Issue" selected. It shows a text input field containing "Not a good topic". At the bottom are "Cancel" and "Confirm" buttons.

Allasian uses cookies to improve your browsing experience, perform analytics and research, and conduct advertising. Accept all cookies to indicate that you agree to our use of cookies on your device. [Allasian cookies and tracking notice](#)

Preferences Only necessary Accept all

Jira Your work Projects Filters Dashboards Teams Plans Apps Create 10 days left Search Start stand-up Complete sprint ...

Experiment 5 Software project

Planning Timeline Backlog Board Calendar List Goals Issues + Add view Development Code You're in a team-managed project Learn more

Projects / Experiment 5 SCRUM Sprint 1

9 days | 10 days left | Start stand-up | Complete sprint | ...

Search AD

GROUP BY None Insights View settings

TO DO 4 IN PROGRESS DONE

Choose a topic SCRM-1 Research on the topic SCRM-2 Implementation SCRM-3 Review SCRM-4

+ Create issue Quickstart

This screenshot shows the Jira SCRUM Sprint 1 board. The board has three columns: TO DO, IN PROGRESS, and DONE. The TO DO column contains four items: 'Choose a topic' (SCRM-1), 'Research on the topic' (SCRM-2), 'Implementation' (SCRM-3), and 'Review' (SCRM-4). The IN PROGRESS and DONE columns are currently empty.

Allasian uses your work Projects Filters Dashboards Teams Plans Apps Create 10 days left Search Start stand-up Complete sprint ...

Experiment 5 Software project

Planning Timeline Backlog Board Calendar List Goals Issues + Add view Development Code You're in a team-managed project Learn more

Projects / Experiment 5 SCRUM Sprint 1

Remove Flag

Issue Choose a topic

Aa B I ... A Aa B I ... Good topic

Cancel Confirm

This screenshot shows a modal dialog titled 'Remove Flag'. It contains a note 'Good topic' and two buttons: 'Cancel' and 'Confirm'.

Jira Your work Projects Filters Dashboards Teams Plans Apps Create 10 days left Search Start stand-up Complete sprint ...

Experiment 5 Software project

Planning Timeline Backlog Board Calendar List Goals Issues + Add view Development Code Project pages Add shortcut You're in a team-managed project Learn more

Projects / Experiment 5 Backlog

Epic

SCRUM Sprint 1 25 Mar – 8 Apr (4 issues)

SCRUM-1 Choose a topic SCRUM-2 Research on the topic SCRUM-3 Implementation SCRUM-4 Review

+ Create issue

Backlog (0 issues)

Your backlog is empty.

+ Create issue Create sprint

Quickstart

This screenshot shows the Jira Backlog view. It displays a summary for 'SCRUM Sprint 1' (25 Mar – 8 Apr) which contains 4 issues: 'SCRUM-1 Choose a topic', 'SCRUM-2 Research on the topic', 'SCRUM-3 Implementation', and 'SCRUM-4 Review'. Below this, it shows an empty backlog with the message 'Your backlog is empty.' and a '+ Create issue' button.

The screenshot shows the Jira Timeline view for the 'Experiment 5' project. The timeline spans from February to May. A blue vertical bar marks the start of 'SCRUM Sprint 1' in March. The interface includes a sidebar with project navigation and a search bar at the top right.

This screenshot is identical to the one above, but it includes a detailed pop-up window for 'SCRUM Sprint 1'. The window shows the sprint is active, has a goal, and specifies a start date of 2025/03/25 and an end date of 2025/04/08.

**Conclusion:**  
Scrum project was implemented using JIRA.

## Experiment 08

### **Aim:**

To Study Project Scheduling Using Gantt chart in ClickUp.

### **Theory:**

#### Project Scheduling

A project schedule is a timeline that outlines the tasks, milestones, deadlines, and resources required for completing a project. It helps project managers organise and plan the sequence of activities, allocate resources efficiently, set realistic timelines, and monitor progress. Having a clear project schedule is crucial for a project manager as it ensures tasks are completed on time, helps in managing resources effectively, allows for better coordination among team members, and assists in identifying potential issues or delays, enabling timely adjustments to keep the project on track.

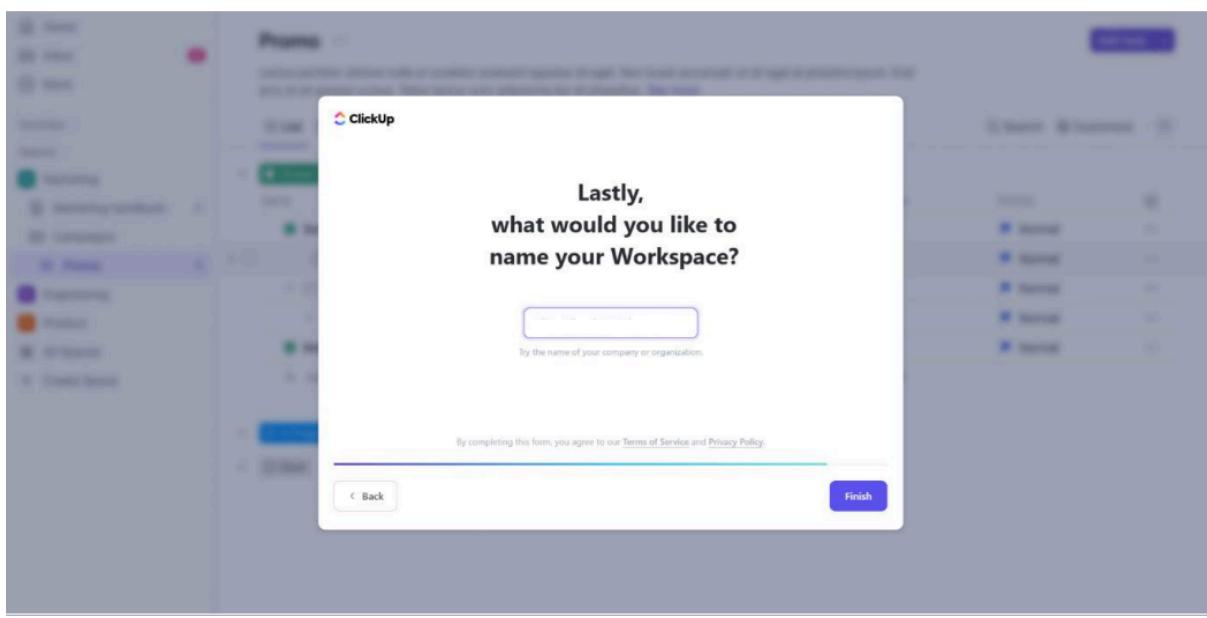
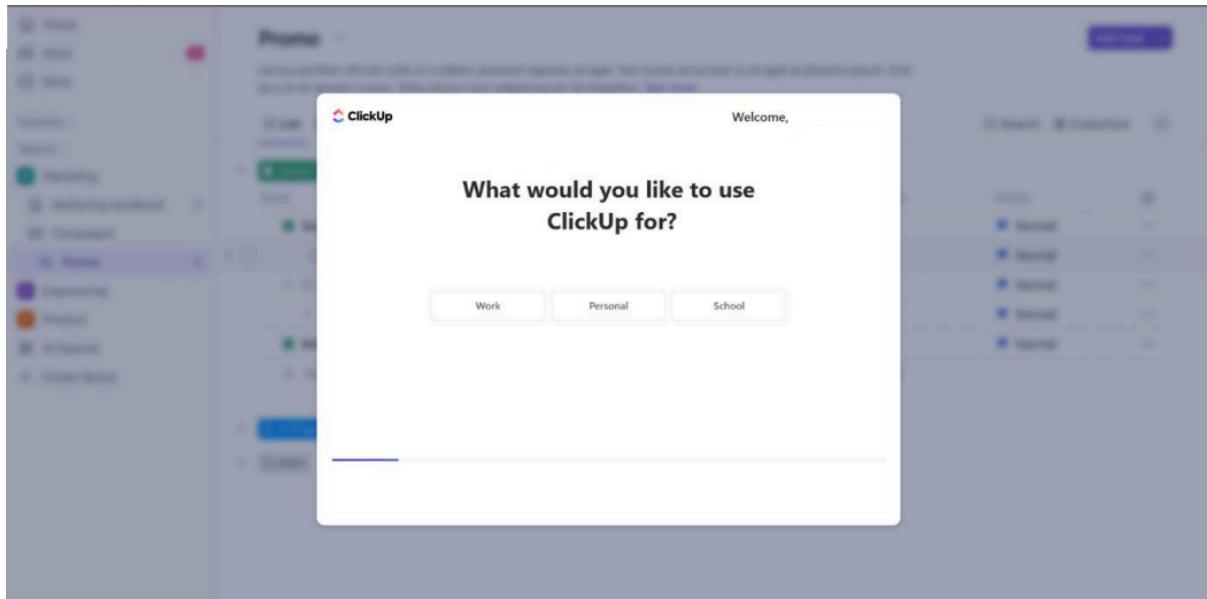
#### Gantt chart

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and along the top is a suitable time scale. Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity.

#### About the topic

It is a Logistic regression Model designed on Agricultural decision making based on soil characteristics and various other environmental factors. Our dataset Encompasses of essential parameters such as soil composition (Nitrogen, potassium and phosphorus contents of the soil as well as the pH level of the soil) and location specific variables such as temperature, humidity and rainfall. Our model leverages these inputs and aims to predicts the most suitable crop for a given soil and environmental conditions.

### **Output:**



Team Space / Projects / Project 1 ...

Board List Calendar Gantt Table + View

Today Week Auto Fit Export

Search... AI Upgrade New Automate Ask AI Share

Search Hide Customize Add Task

Sort Filter Mode Assignee Search...

Mar 2 - 8 Mar 9 - 15 Mar 16 - 22 Mar 23 - 29

Project 1

- Acquiring the topic
- Topic Research
- Designing the logistic regression model
- Examination break
- Model implementation
- Final touchups

Getting started 0% complete

- Understand the ClickUp Hierarchy
- Manage your work with tasks
- Customize your views
- Collaborate with your team via Inbox

This view has unsaved changes. Save to update this view for everyone.

Revert Enable Automate Save CTA

This screenshot shows the ClickUp Gantt chart view for a project named "Project 1". The chart displays tasks from March 2 to 29, 2024. The tasks are: "Acquiring the topic" (Mar 2-4), "Topic Research" (Mar 5-9), "Designing the logistic regression model" (Mar 9-15), "Examination break" (Mar 15-16), "Model implementation" (Mar 16-22), and "Final touchups" (Mar 23-29). A "Getting started" sidebar is visible on the left, and a message at the bottom right indicates unsaved changes.

Three screenshots of a project management application interface, showing different tasks and their details.

**Screenshot 1: Acquiring the topic**

**Task Details:**

- Status: COMPLETE
- Dates: Today → Today
- Time Estimate: Empty
- Tags: Empty
- Assignees: [Empty]
- Priority: Empty
- Track Time: Empty
- Relationships: Empty

**Activity Log:**

- Dharitha Iewani changed status from To Do to Complete 6 mins ago

**Screenshot 2: Topic Research**

**Task Details:**

- Status: COMPLETE
- Dates: Tomorrow → Sat
- Time Estimate: Empty
- Tags: Empty
- Assignees: [Empty]
- Priority: Empty
- Track Time: Empty
- Relationships: Empty

**Activity Log:**

- You changed due date from Fri to Sat 5 mins ago

**Screenshot 3: Designing the logistic regression model**

**Task Details:**

- Status: COMPLETE
- Dates: Sun → Tue
- Time Estimate: Empty
- Tags: Empty
- Assignees: [Empty]
- Priority: Empty
- Track Time: Empty
- Relationships: Empty

**Activity Log:**

- Dharitha Iewani changed status from To Do to Complete 4 mins ago

The screenshot shows a task creation interface. The title is "Examination break". The status is set to "PENDING". Dates are listed as "Mar 12" and "Mar 15". Priority is "Low". The task is assigned to two users. There is no time estimate or tag. Relationships are empty. A note at the top says "Ask Brain to write a description - generate subtasks - find similar tasks - or ask about this task".

**Activity**

- Show more
- Dharini Jaiswal changed status from To Do to Complete 3 mins

The screenshot shows a task creation interface. The title is "Model implementation". The status is "COMPLETE". Dates are "Mar 16" and "Mar 19". Priority is "Low". The task is assigned to two users. There is no time estimate or tag. Relationships are empty. A note at the top says "Ask Brain to write a description - generate subtasks - find similar tasks - or ask about this task".

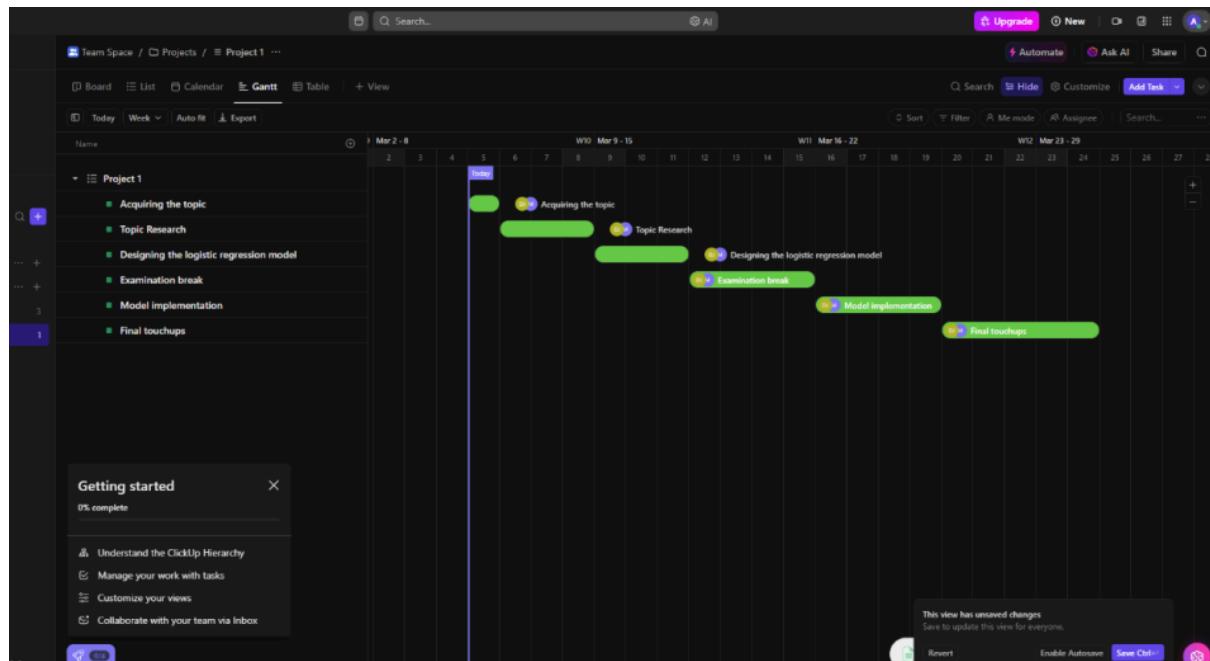
**Activity**

- Show more
- Dharini Jaiswal changed status from To Do to Complete 3 mins

The screenshot shows a task creation interface. The title is "Final touchups". The status is "COMPLETE". Dates are "Mar 20" and "Mar 24". Priority is "Low". The task is assigned to two users. There is no time estimate or tag. Relationships are empty. A note at the top says "Ask Brain to write a description - generate subtasks - find similar tasks - or ask about this task".

**Activity**

- Show more
- Dharini Jaiswal changed status from To Do to Complete 2 mins



## Conclusion:

This project delved into the application of polynomial regression in financial data analysis, comparing implementations from scratch using Python libraries with those utilizing PyTorch. Through data preprocessing, model implementation, training, and evaluation, we examined the performance of each approach. While both methods proved effective, PyTorch demonstrated advantages in computational efficiency and scalability. This project underscores the importance of selecting appropriate tools and frameworks based on the task's requirements, offering insights into the practical utility of polynomial regression in financial modeling.

## **Software Engineering & Project Management Lab Experiment No: - 09**

**Aim: To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers**

### **Theory:**

#### **What is Docker?**

Docker is a platform that allows developers to build, package, and deploy applications in lightweight, portable containers. These containers include everything needed to run an application, such as code, runtime, system tools, libraries, and dependencies.

#### **Containerization Technology**

Containers are isolated environments where applications run independently. Unlike traditional virtualization, which requires separate operating systems for each application, containers share the host OS kernel, making them faster and more efficient.

#### **Docker Architecture**

Docker follows a client-server architecture consisting of the following key components:

##### **1. Docker Client**

- It is the command-line interface (CLI) that allows users to interact with Docker.
- Commands such as docker run, docker build, and docker stop are executed through the client.

##### **2. Docker Daemon (dockerd)**

- It runs in the background and manages Docker containers, images, volumes, and networks.
- It listens for requests from the Docker Client and executes commands.

##### **3. Docker Images**

- A Docker image is a read-only template containing the application code, libraries, and dependencies.
- Images are created using Dockerfiles, which define the steps to build an image.
- Images are stored in Docker Hub or private repositories.

## **Software Engineering & Project Management Lab Experiment No: - 09**

**Aim: To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers**

### **4. Docker Containers**

- A container is an instance of a Docker image running as an isolated process on a host machine.
- Containers are lightweight, portable, and can be started, stopped, or removed as needed.

### **5. Docker Registry**

- It is a storage system for Docker images.
- The public registry, Docker Hub, provides access to a vast collection of pre-built images.
- Users can also create private registries for security and control.

## **Docker Container Life Cycle**

The **life cycle of a container** follows these steps:

1. **Create** – A container is created from an image using the docker create command.
2. **Start** – The container starts running using the docker start command.
3. **Run** – A new container can be started directly using docker run.
4. **Pause/Unpause** – Containers can be temporarily paused and resumed.
5. **Stop** – The container can be stopped using docker stop.
6. **Restart** – A stopped container can be restarted.
7. **Kill** – A container can be forcefully stopped using docker kill.
8. **Remove** – Containers that are no longer needed can be deleted using docker rm.

## **Benefits of Docker**

### **1. Portability**

- Containers can run on any platform that supports Docker.
- Applications behave consistently across different environments.

### **2. Efficiency**

- Containers share the host OS kernel, reducing overhead and improving performance.
- They consume fewer resources compared to virtual machines.

## **Software Engineering & Project Management Lab Experiment No: - 09**

**Aim: To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers**

### **3. Isolation**

- Each container runs in its own isolated environment, preventing dependency conflicts.

### **4. Scalability**

- Applications can be scaled up quickly by launching multiple containers.
- Docker enables automatic load balancing in large-scale deployments.

### **5. Consistency**

- Ensures that the application runs the same way in development, testing, and production.
- Eliminates the "works on my machine" problem.

### **Docker Engine:**

At the core of Docker is the Docker Engine, which is responsible for building, running, and managing containers. It consists of the Docker daemon, which manages containers, images, networks, and volumes, and the Docker client, which allows users to interact with the daemon through the Docker API.

### **Docker Images:**

Docker images are read-only templates used to create containers. They contain the application code, runtime, libraries, dependencies, and other files needed to run the application. Images are built using Dockerfiles, which are text files that define the steps needed to create the image.

## Software Engineering & Project Management Lab Experiment No: - 09

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers

### OUTPUT:-

```
C:\Users\202>docker run redis
Unable to find image 'redis:latest' locally
latest: Pulling from library/redis
8a1e25ce7c4f: Pull complete
8ab039a68e51: Pull complete
2b12a49dcfb9: Pull complete
cdf9868f47ac: Pull complete
e73ea5d3136b: Pull complete
890ad32c613f: Pull complete
4f4fb700ef54: Pull complete
ba517b76f92b: Pull complete
Digest: sha256:7dd707032d90c6eaafdf566f62a00f5b0116ae08fd7d6cbbb0f311b82b47171a2
Status: Downloaded newer image for redis:latest
1:C 13 Mar 2024 03:19:03.928 * o000o000o000o Redis is starting o000o000o000o
1:C 13 Mar 2024 03:19:03.928 * Redis version=7.2.4, bits=64, commit=00000000, mod
1:C 13 Mar 2024 03:19:03.928 # Warning: no config file specified, using the defau
s.conf
1:M 13 Mar 2024 03:19:03.929 * monotonic clock: POSIX clock_gettime
1:M 13 Mar 2024 03:19:03.929 * Running mode=standalone, port=6379.
1:M 13 Mar 2024 03:19:03.929 * Server initialized
1:M 13 Mar 2024 03:19:03.929 * Ready to accept connections tcp
1:signal-handler (1710300105) Received SIGINT scheduling shutdown...
1:M 13 Mar 2024 03:21:45.877 * User requested shutdown...
1:M 13 Mar 2024 03:21:45.877 * Saving the final RDB snapshot before exiting.
1:M 13 Mar 2024 03:21:45.887 * DB saved on disk
1:M 13 Mar 2024 03:21:45.887 # Redis is now ready to exit, bye bye...
```

C:\Users\202>docker images				
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
redis	latest	170a1e90f843	2 months ago	138MB

```
C:\Users\202>docker pull redis
Using default tag: latest
latest: Pulling from library/redis
Digest: sha256:7dd707032d90c6eaafdf566f62a00f5b0116ae08fd7d6cbbb0f311b82b47171a2
Status: Image is up to date for redis:latest
docker.io/library/redis:latest
```

C:\Users\202>docker ps						
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
052aecb0ee88	redis	"docker-entrypoint.s..."	About a minute ago	Up 4 seconds	6379/tcp	container121
1c4472744083	redis	"docker-entrypoint.s..."	6 minutes ago	Up 10 seconds	6379/tcp	modest_herschel

## Software Engineering & Project Management Lab Experiment No: - 09

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers

```
C:\Users\202>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
052aecb0ee88 redis "docker-entrypoint.s..." About a minute ago Up 4 seconds 6379/tcp container121
1c4472744083 redis "docker-entrypoint.s..." 6 minutes ago Up 10 seconds 6379/tcp modest_herschel
```

```
C:\Users\202>
C:\Users\202>
C:\Users\202>docker stop 052aecb0ee88
052aecb0ee88
```

```
C:\Users\202>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
1c4472744083 redis "docker-entrypoint.s..." 9 minutes ago Up 2 minutes 6379/tcp modest_herschel
```

```
C:\Users\202>docker start 052aecb0ee88
052aecb0ee88
```

```
C:\Users\202>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
052aecb0ee88 redis "docker-entrypoint.s..." 5 minutes ago Up 8 seconds 6379/tcp container121
1c4472744083 redis "docker-entrypoint.s..." 10 minutes ago Up 3 minutes 6379/tcp modest_herschel
```

```
C:\Users\202>docker rm 052aecb0ee88
052aecb0ee88
```

```
C:\Users\202>docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
redis latest 170ale90f843 2 months ago 138MB
```

```
C:\Users\202>
```

```
C:\Users\202>docker exec -d 1c4472744083 touch /tmp/execWorks
```

```
C:\Users\202>docker exec -it 1c4472744083 bash
root@1c4472744083:/data# |
```

```
C:\Users\202>docker restart 1c4472744083
1c4472744083
```

```
C:\Users\202>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
1c4472744083 redis "docker-entrypoint.s..." 13 minutes ago Up 3 seconds 6379/tcp modest_herschel
```

## Software Engineering & Project Management Lab Experiment No: - 09

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers

```
C:\Users\202>docker inspect 1c4472744083
[
  {
    "Id": "1c44727440831475b093dcf93163064b819bdd9ad8378bb3a4fa847dc411d80",
    "Created": "2024-03-13T03:19:03.418741433Z",
    "Path": "docker-entrypoint.sh",
    "Args": [
      "redis-server"
    ],
    "State": {
      "Status": "running",
      "Running": true,
      "Paused": false,
      "Restarting": false,
      "OOMKilled": false,
      "Dead": false,
      "Pid": 2112,
      "ExitCode": 0,
      "Error": "",
      "StartedAt": "2024-03-13T03:32:13.750463204Z",
      "FinishedAt": "2024-03-13T03:32:13.145321277Z"
    },
    "Image": "sha256:170a1e90f8436daa6778a3926e716928826c215ca23a8df8055f663f9428",
    "ResolvConfPath": "/var/lib/docker/containers/1c44727440831475b093dcf93163064b819bdd9a
}
```

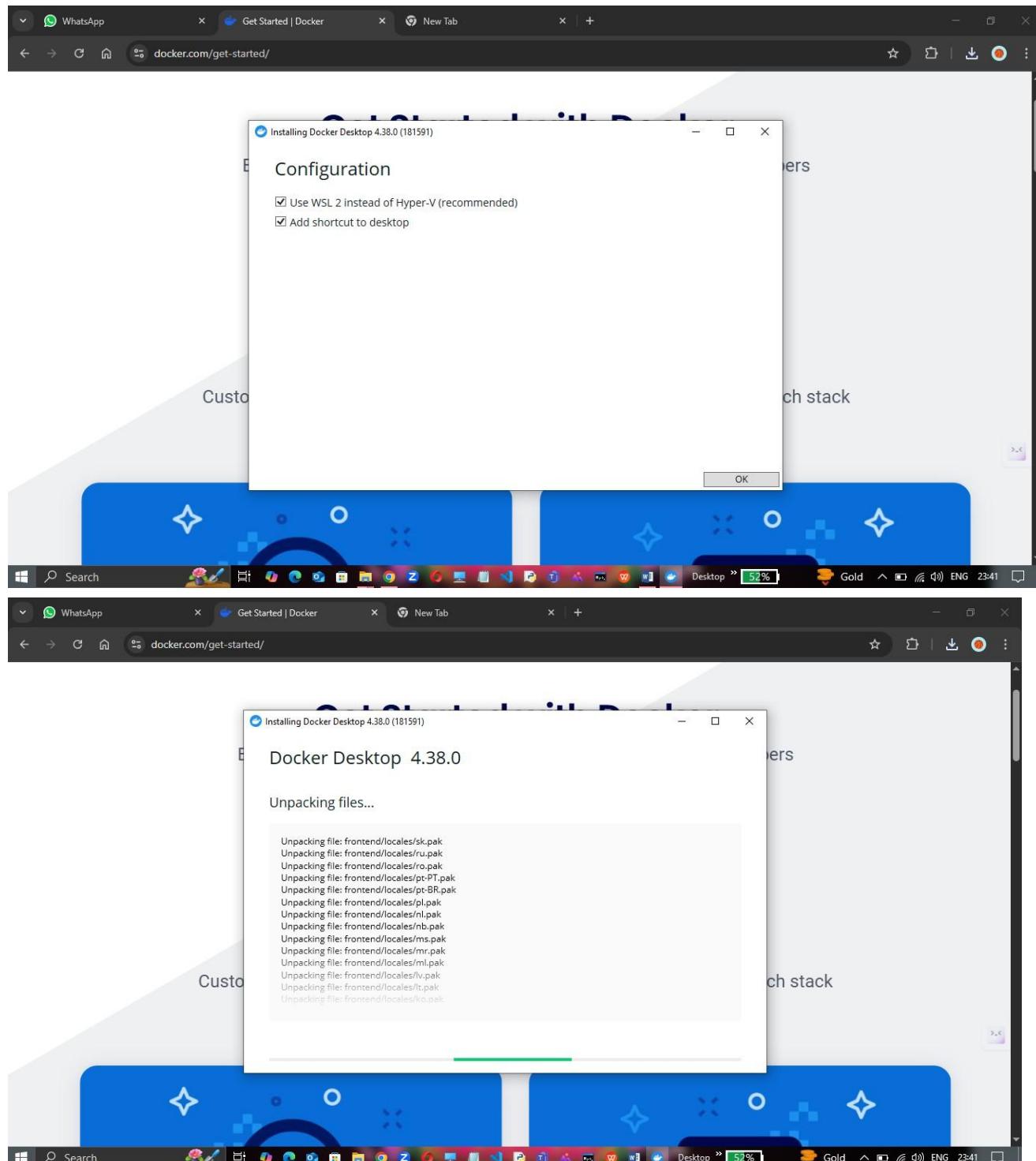
```
C:\Users\202>docker commit 1c4472744083 new_image_name:redis2
sha256:33e4284a7e92a4a1331555d01f6e078fc496e3a3ed8eb7f84f2678261ad07e83

C:\Users\202>docker images
REPOSITORY          TAG           IMAGE ID        CREATED         SIZE
new_image_name     redis2        33e4284a7e92   4 seconds ago  138MB
new_image_name     tag          61ab016507fa   36 seconds ago  138MB
redis              latest        170a1e90f843   2 months ago   138MB
```

## Software Engineering & Project Management Lab Experiment No: - 09

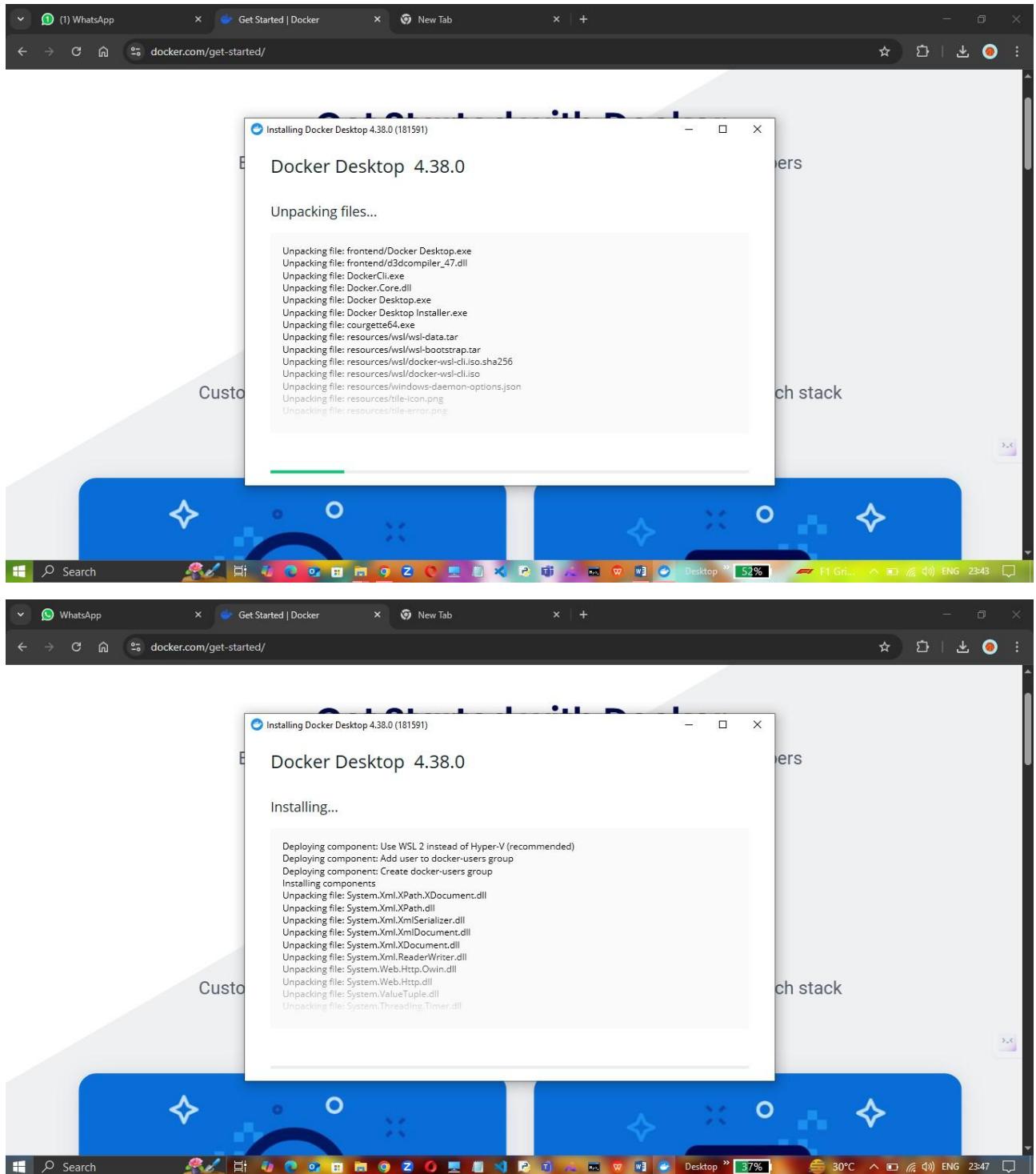
**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers

### SCREENSHOTS:



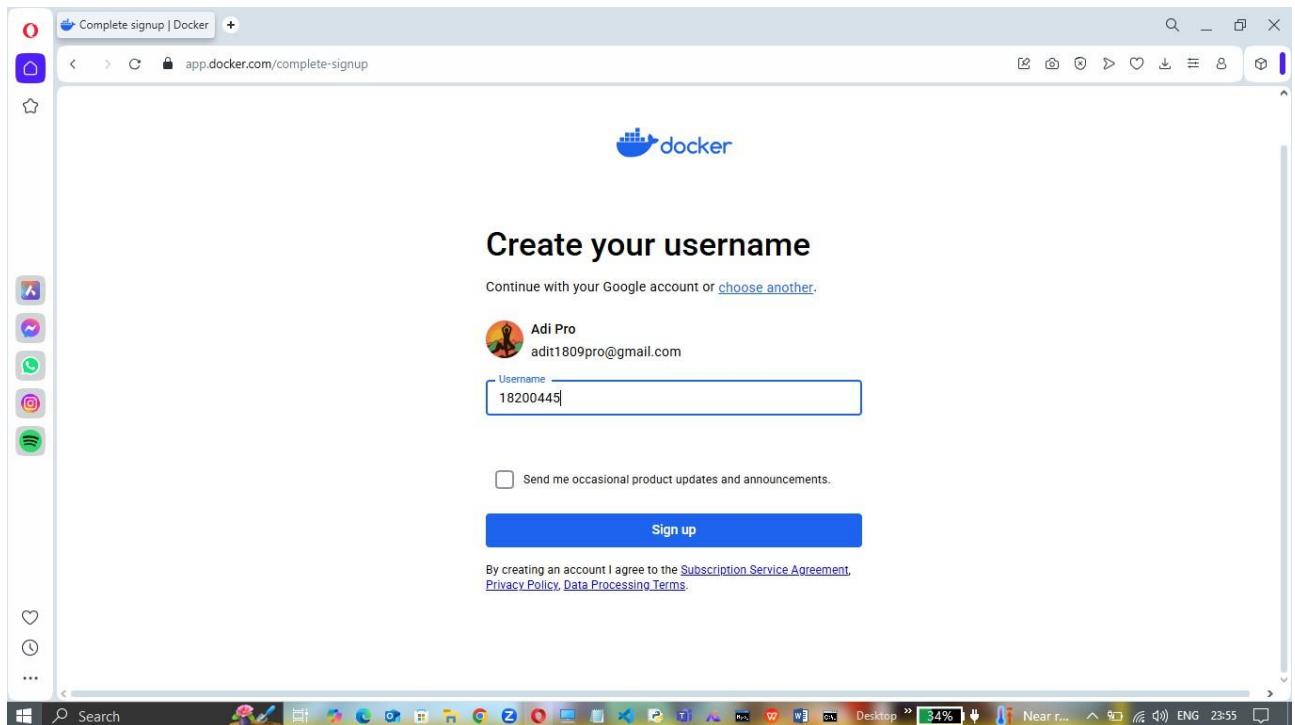
## Software Engineering & Project Management Lab Experiment No: - 09

**Aim: To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers**



## Software Engineering & Project Management Lab Experiment No: - 09

**Aim: To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers**

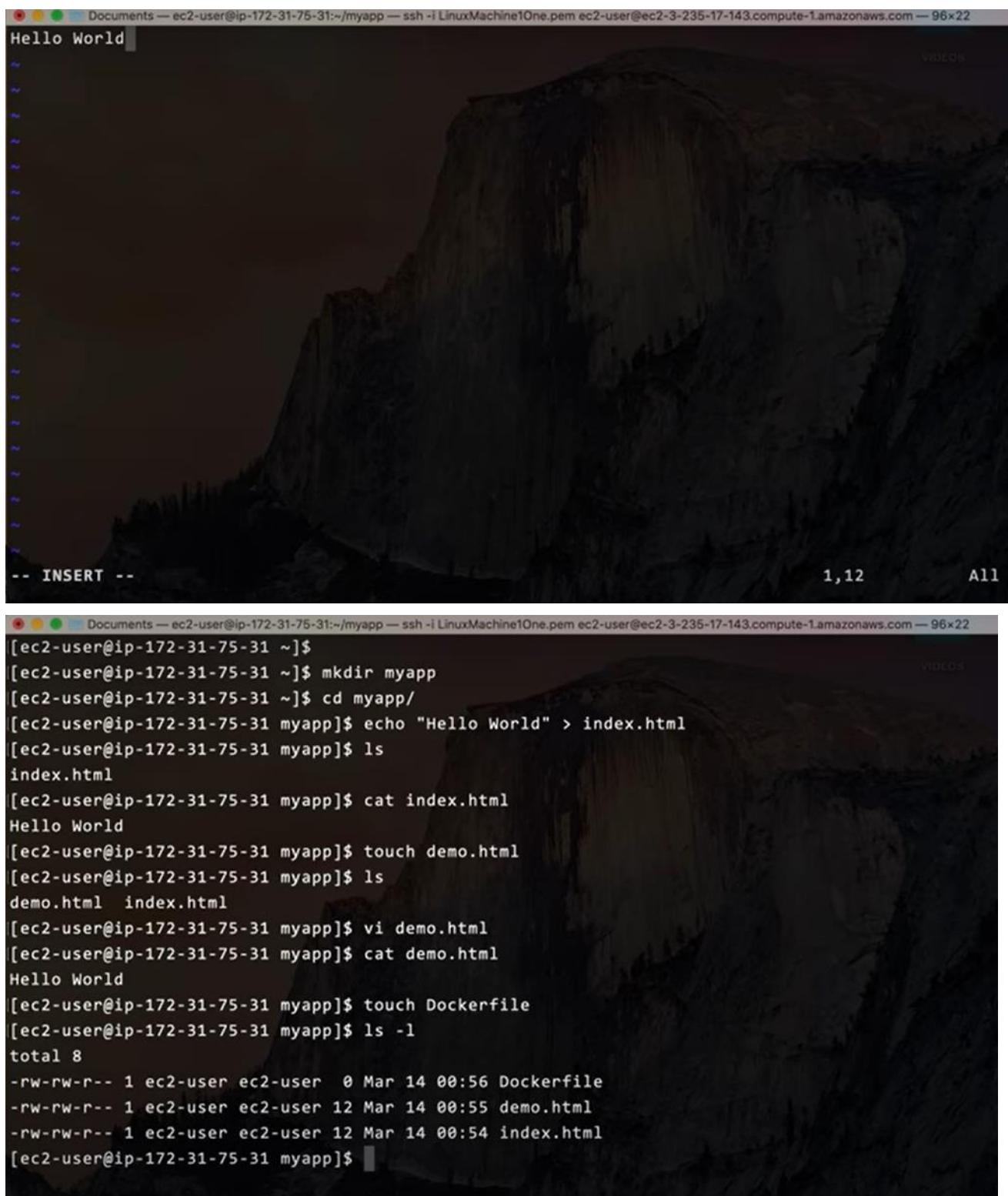
A screenshot of a terminal window titled "Documents — ec2-user@ip-172-31-75-31 ~]\$. ssh -i LinuxMachine1One.pem ec2-user@ec2-3-235-17-143.compute-1.amazonaws.com — 96x22". The terminal history shows the following commands:

```
[ec2-user@ip-172-31-75-31 ~]$  
[ec2-user@ip-172-31-75-31 ~]$ mkdir myapp  
[ec2-user@ip-172-31-75-31 ~]$ cd myapp/  
[ec2-user@ip-172-31-75-31 myapp]$ echo "Hello World" > index.html  
[ec2-user@ip-172-31-75-31 myapp]$ ls  
index.html  
[ec2-user@ip-172-31-75-31 myapp]$ cat index.html  
Hello World  
[ec2-user@ip-172-31-75-31 myapp]$ touch demo.html  
[ec2-user@ip-172-31-75-31 myapp]$ ls  
demo.html index.html  
[ec2-user@ip-172-31-75-31 myapp]$ vi demo.html
```

The background of the terminal window features a dark landscape image.

## Software Engineering & Project Management Lab Experiment No: - 09

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers



The screenshot shows a terminal window with a dark background featuring a mountain landscape. The terminal title bar reads "Documents — ec2-user@ip-172-31-75-31:~/myapp — ssh -i LinuxMachine1One.pem ec2-user@ec2-3-235-17-143.compute-1.amazonaws.com — 96x22". The terminal content is as follows:

```
[ec2-user@ip-172-31-75-31 ~]$  
[ec2-user@ip-172-31-75-31 ~]$ mkdir myapp  
[ec2-user@ip-172-31-75-31 ~]$ cd myapp/  
[ec2-user@ip-172-31-75-31 myapp]$ echo "Hello World" > index.html  
[ec2-user@ip-172-31-75-31 myapp]$ ls  
index.html  
[ec2-user@ip-172-31-75-31 myapp]$ cat index.html  
Hello World  
[ec2-user@ip-172-31-75-31 myapp]$ touch demo.html  
[ec2-user@ip-172-31-75-31 myapp]$ ls  
demo.html index.html  
[ec2-user@ip-172-31-75-31 myapp]$ vi demo.html  
[ec2-user@ip-172-31-75-31 myapp]$ cat demo.html  
Hello World  
[ec2-user@ip-172-31-75-31 myapp]$ touch Dockerfile  
[ec2-user@ip-172-31-75-31 myapp]$ ls -l  
total 8  
-rw-rw-r-- 1 ec2-user ec2-user 0 Mar 14 00:56 Dockerfile  
-rw-rw-r-- 1 ec2-user ec2-user 12 Mar 14 00:55 demo.html  
-rw-rw-r-- 1 ec2-user ec2-user 12 Mar 14 00:54 index.html  
[ec2-user@ip-172-31-75-31 myapp]$
```

## Software Engineering & Project Management Lab Experiment No: - 09

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers

```
Documents — ec2-user@ip-172-31-75-31:~/myapp — ssh -i LinuxMachine1One.pem ec2-user@ec2-3-235-17-143.compute-1.amazonaws.com — 96x22
[ec2-user@ip-172-31-75-31 myapp]$ ls
Dockerfile  demo.html  index.html
[ec2-user@ip-172-31-75-31 myapp]$ vi Dockerfile
```

```
FROM nginx
COPY index.html /usr/share/nginx/html
```

-- INSERT --

2,38

All

## Software Engineering & Project Management Lab Experiment No: - 09

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers

```
Documents — ec2-user@ip-172-31-75-31:~/myapp — ssh -i LinuxMachine1One.pem ec2-user@ec2-3-235-17-143.compute-1.amazonaws.com — 96x22
[ec2-user@ip-172-31-75-31 myapp]$
[ec2-user@ip-172-31-75-31 myapp]$ ls
Dockerfile demo.html index.html
[ec2-user@ip-172-31-75-31 myapp]$ vi Dockerfile
[ec2-user@ip-172-31-75-31 myapp]$ cat Dockerfile
FROM nginx
COPY index.html /usr/share/nginx/html
[ec2-user@ip-172-31-75-31 myapp]$ docker info
Client:
  Context:    default
  Debug Mode: false

Server:
ERROR: Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?
errors pretty printing info
[ec2-user@ip-172-31-75-31 myapp]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-75-31 myapp]$
```

```
Documents — ec2-user@ip-172-31-75-31:~/myapp — ssh -i LinuxMachine1One.pem ec2-user@ec2-3-235-17-143.compute-1.amazonaws.com — 96x22
init version: de40ad0
Security Options:
  seccomp
    Profile: default
Kernel Version: 5.10.167-147.601.amzn2.x86_64
Operating System: Amazon Linux 2
OSType: linux
Architecture: x86_64
CPUs: 1
Total Memory: 964.8MiB
Name: ip-172-31-75-31.ec2.internal
ID: 3DRI:26BR:Y5X4:GCJ2:2UYQ:FHFW:AQ5Q:SUIY:67Z2:VVGE:KC6M:DHX2
Docker Root Dir: /var/lib/docker
Debug Mode: false
Registry: https://index.docker.io/v1/
Labels:
Experimental: false
Insecure Registries:
  127.0.0.0/8
Live Restore Enabled: false

[ec2-user@ip-172-31-75-31 myapp]$
```

## Software Engineering & Project Management Lab Experiment No: - 09

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers

```
Documents — ec2-user@ip-172-31-75-31:~/myapp — ssh -i LinuxMachine1One.pem ec2-user@ec2-3-235-17-143.compute-1.amazonaws.com — 96x22
[ec2-user@ip-172-31-75-31 myapp]$
[ec2-user@ip-172-31-75-31 myapp]$ docker build -t myapp .
Sending build context to Docker daemon 4.096kB
Step 1/2 : FROM nginx
--> 904b8cb13b93
Step 2/2 : COPY index.html /usr/share/nginx/html
--> dffa39f040c6
Successfully built dffa39f040c6
Successfully tagged myapp:latest
[ec2-user@ip-172-31-75-31 myapp]$ docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
myapp           latest        dffa39f040c6   25 seconds ago  142MB
nginx           latest        904b8cb13b93   12 days ago    142MB
hello-world     latest        feb5d9fea6a5   17 months ago   13.3kB
[ec2-user@ip-172-31-75-31 myapp]$
[ec2-user@ip-172-31-75-31 myapp]$ docker run -p 8080:80 myapp
```

```
Documents — ec2-user@ip-172-31-75-31:~/myapp — ssh -i LinuxMachine1One.pem ec2-user@ec2-3-235-17-143.compute-1.amazonaws.com — 96x22
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
myapp           latest        dffa39f040c6   25 seconds ago  142MB
nginx           latest        904b8cb13b93   12 days ago    142MB
hello-world     latest        feb5d9fea6a5   17 months ago   13.3kB
[ec2-user@ip-172-31-75-31 myapp]$
[ec2-user@ip-172-31-75-31 myapp]$ docker run -p 8080:80 myapp
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/03/14 01:03:25 [notice] 1#1: using the "epoll" event method
2023/03/14 01:03:25 [notice] 1#1: nginx/1.23.3
2023/03/14 01:03:25 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/03/14 01:03:25 [notice] 1#1: OS: Linux 5.10.167-147.601.amzn2.x86_64
2023/03/14 01:03:25 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 32768:65536
2023/03/14 01:03:25 [notice] 1#1: start worker processes
2023/03/14 01:03:25 [notice] 1#1: start worker process 29
```

## Software Engineering & Project Management Lab Experiment No: - 09

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers

```
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/03/14 01:03:25 [notice] 1#1: using the "epoll" event method
2023/03/14 01:03:25 [notice] 1#1: nginx/1.23.3
2023/03/14 01:03:25 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/03/14 01:03:25 [notice] 1#1: OS: Linux 5.10.167-147.601.amzn2.x86_64
2023/03/14 01:03:25 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 32768:65536
2023/03/14 01:03:25 [notice] 1#1: start worker processes
2023/03/14 01:03:25 [notice] 1#1: start worker process 29
^C2023/03/14 01:03:47 [notice] 1#1: signal 2 (SIGINT) received, exiting
2023/03/14 01:03:47 [notice] 29#29: exiting
2023/03/14 01:03:47 [notice] 29#29: exit
2023/03/14 01:03:47 [notice] 1#1: signal 17 (SIGCHLD) received from 29
2023/03/14 01:03:47 [notice] 1#1: worker process 29 exited with code 0
2023/03/14 01:03:47 [notice] 1#1: exit
[ec2-user@ip-172-31-75-31 myapp]$
```

```
Documents — ec2-user@ip-172-31-75-31:~/myapp — ssh -i LinuxMachine1One.pem ec2-user@ec2-3-235-17-143.compute-1.amazonaws.com — 96x22
[ec2-user@ip-172-31-75-31 myapp]$
[ec2-user@ip-172-31-75-31 myapp]$ docker run -d -p 8080:80 myapp
f31fe21f8fc1a77ee768f2604ab695bc8e87733d95a587a62b482c3cd9fa11e6
[ec2-user@ip-172-31-75-31 myapp]$
[ec2-user@ip-172-31-75-31 myapp]$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED             STATUS              PORTS
NAMES
f31fe21f8fc1   myapp      "/docker-entrypoint..."   7 seconds ago    Up 6 seconds   0.0.0.0:8080->80
0/tcp, :::8080->80/tcp   ecstatic_beaver
[ec2-user@ip-172-31-75-31 myapp]$
```

**Conclusion:** Thus, we have successfully installed Docker and execute docker commands to manage images and interact with containers.

## **Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

Name : Aarya Thakur

T22

Roll No: 114

### **Theory:**

#### **What is Docker?**

Docker is a platform that allows developers to build, package, and deploy applications in lightweight, portable containers. These containers include everything needed to run an application, such as code, runtime, system tools, libraries, and dependencies.

#### **Benefits of Docker**

##### **1. Portability**

- Containers can run on any platform that supports Docker.
- Applications behave consistently across different environments.

##### **2. Efficiency**

- Containers share the host OS kernel, reducing overhead and improving performance.
- They consume fewer resources compared to virtual machines.

##### **3. Isolation**

- Each container runs in its own isolated environment, preventing dependency conflicts.

##### **4. Scalability**

- Applications can be scaled up quickly by launching multiple containers.
- Docker enables automatic load balancing in large-scale deployments.

##### **5. Consistency**

- Ensures that the application runs the same way in development, testing, and production.

## **Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

- Eliminates the "works on my machine" problem.

### **DIRECTIVE argument**

Although the DIRECTIVE is case-insensitive, it is recommended to write all directives in uppercase to differentiate them from arguments. A Dockerfile usually consists of multiple lines of instructions that are executed sequentially by the Docker engine during the image-building process.

## **Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

Now that you understand the purpose of a Dockerfile, let's get our hands dirty by building one for a sample Python application.

### **Building a Dockerfile for a Sample Python/Flask Application**

The application we'll be working with is a simple Flask app with only one home route that returns Hello, World!.

Let's start by setting up the Flask application. Open your favorite code editor and create a new directory for your project. In this directory, create a new file named app.py and add the following Python code:

```
from flask import Flask  
  
app = Flask(__name__)  
  
@app.route('/')  
  
def hello():  
  
    return 'Hello, World!'
```

### **Now, let's build the Dockerfile.**

In the same directory as your app.py, create a new file named Dockerfile (with no file extension). This is where you'll write the instructions for Docker to build your image.

Now, follow the steps below to create the Dockerfile:

#### **#1: Specify the base image**

The very first instruction you write in a Dockerfile must be the FROM directive, which specifies the base image. The base image is the image from which all other layers in your Docker image will be built. It's the foundation of your Docker image, much like the foundation of a building.

Add the following instruction to your Dockerfile:

```
FROM python:3.11-slim
```

## **Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

Here, we're telling Docker to use the official Python Docker image, and more specifically, the 3.11-slim version. This slim variant of Python Docker image is a minimal version that excludes some packages to make the image smaller. Note that the base image you specify will be downloaded from Docker Hub, Docker's official image registry. The reason we're using Python as the base image is that the application containerization is written in Python. The Python base image includes a Python interpreter, which is necessary to run Python code, as well as a number of commonly used Python utilities and libraries. By using the Python base image, we're ensuring that the environment within the Docker container is preconfigured for running Python code.

### **#2 Set the working directory**

Once you've chosen the base image, the next step is to determine the working directory using the WORKDIR directive. Insert the following line after the FROM directive:

```
WORKDIR /app
```

Here, we're telling Docker to create a directory named app in the Docker container and use it as the current working directory. All instructions that FROM python:3.11-slim WORKDIR /app follow (like RUN, COPY, and CMD) will be run in this directory inside the container. Think of this as typing the command cd /app in a terminal to change the current working directory to /app. The difference here is that it's being done within the Docker container as part of the build process. A working directory within the container is necessary because it designates a specific location for our application code within the container and determines where commands will be run from. If we don't set a working directory, Docker won't have a clear context for where your application is located, which would make it harder to interact with.

**#3 Install dependencies** Once the working directory is set, the next step is to install the dependencies. Our Python application relies on the Flask web framework, which manages requests, routes URLs, and handles other web-related tasks. To install Flask, add the following instruction in your Dockerfile just under the WORKDIR directive:

**Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

**RUN pip install flask==2.3**

## **Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

Here, we're instructing Docker to use pip (a package installer for Python) to install the specific version of Flask we need for our application.

**#4 Copy application files to the container** After setting up the working directory and installing the necessary dependencies, we're now ready to copy the application files into the Docker container. To do this, add the following instruction just below the RUN directive:

```
COPY . /app
```

This line copies everything in the current directory (denoted by ".") on our host machine into the /app directory we previously set as our working directory within the Docker container. It's like using the cp command in the terminal to copy files from one directory to another, but in this context, it's copying files from your local machine to the Docker container. Why do we need to do this? It's simple. Without this step, the Docker container wouldn't have access to our application's code, making it impossible to run our app.

**#5 Specify the environment variable** Once the application files are copied, we need to set up the FLASK\_APP environment variable for our Docker container using the ENV directive. Now, you may be wondering why we need this environment variable in the first place. In our app.py file, we create an instance of the Flask application and assign it to the variable app. This application instance is what Flask needs to run, and it's located in the app.py file. When starting our Flask application using the flask run command (which we'll discuss in the next section), Flask must know where to locate the application instance to run. Flask uses the FLASK\_APP environment variable to find this instance. Hence, we need to use the ENV directive to set the value of FLASK\_APP to app.py. To do this, add the following line under the RUN directive:

```
ENV FLASK_APP=app.py
```

This line ensures Flask knows exactly where to find the application instance to run, which in our case is app.py.

## **Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

**#6 Define the default command** The last instruction that we need for our application is to specify the default command that will be executed when the Docker container starts:

```
CMD ["flask", "run", "--host=0.0.0.0", "--port=5000"]
```

from the image, we'll build from this Dockerfile.

**Insert the following instruction below the ENV directive:**

Here's what each part of the argument passed to the CMD directive does:

- flask: This is the program that we want to run. In this case, it's the Flask command-line interface.
- run: This command instructs Flask to start a local development server.
- --host=0.0.0.0: This argument tells the Flask server to listen on all public IPs. In the context of Docker, this means the Flask application will be accessible on any IP address that can reach the Docker container.
- --port=5000: This argument specifies the port number that the Flask server will listen on. Port 5000 is the default port for Flask, but it's good practice to explicitly declare it for clarity.

After this, our Dockerfile is ready. It should look like this:

```
FROM python:3.11-slim
WORKDIR /app
RUN pip install flask==2.3
COPY . /app
ENV FLASK_APP=app.py
CMD ["flask", "run", "--host=0.0.0.0", "--port=5000"]
```

It's worth noting that the directives we used in the Dockerfile for our Python app aren't the only ones available in Docker. But they are the ones you'll often encounter when working with Dockerfiles.

## **Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

**#7 Create a .dockerignore file** Before we go ahead and build our Docker image, we need to take care of one last thing. Remember the following COPY directive?

```
COPY . /app
```

This line instructs Docker to copy everything from our current directory to the app directory inside the container, which includes the Dockerfile itself. But, the Dockerfile isn't required for our app to work—it's just for us to create the Docker image. So, we need to ensure that the Dockerfile doesn't get copied to the app directory in the container. Here's how we do it: Create a new file called .dockerignore in the same directory as your Dockerfile. This file works much like a .gitignore file if you're familiar with Git. Then, add the word Dockerfile to this file. This tells Docker to ignore the Dockerfile when copying files into the container. Now that we've prepared everything, it's time to build our Docker image, run a container from this image, and test our application to see if everything works as expected.

### **Building and running the Docker Image**

Open a terminal and navigate to the directory where your Dockerfile is located. Now, run the following command to create an image named sample-flask-app:v1 (you can name the image anything you prefer):

```
$ docker build . -t sample-flask-app:v1
```

In the command above, the dot (.) after the build command indicates that the current directory is the build context. We're using the -t flag to tag the Docker image with the name sample-flask-app and version v1. After running this command, you'll see an output similar to this:

## Software Engineering & Project Management Lab Experiment No: - 10

**Aim:** To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.

```
$ docker build . -t sample-flask-app:v1
[+] Building 17.7s (10/10) FINISHED docker:default
=> [internal] load build definition from Dockerfile          0.0s
=> => transferring dockerfile: 192B                          0.0s
=> [internal] load metadata for docker.io/library/python:3.11 3.5s
=> [auth] library/python:pull token for registry             0.0s
=> [internal] load .dockerignore                            0.0s
=> => transferring context: 50B                           0.0s
=> [1/4] FROM docker.io/library/python:3.11                9.0s
=> => resolve docker.io/library/python:3.11               0.0s
=> => sha256:1103112ebfc46e01c0f 3.51MB / 3.51MB        1.9s
=> => sha256:b4b80ef7128d12d8dc9bd114 12.87MB / 12.87MB 2.7s
=> => sha256:a2eb07f336e4f1 1.65kB / 1.65kB            0.0s
=> => sha256:4bcd5d5bc81ca 1.37kB / 1.37kB            0.0s
=> => sha256:15646a3fa12dde 6.93kB / 6.93kB            0.0s
=> => sha256:8a1e25ce7c4f 29.12MB / 29.12MB           4.8s
=> => sha256:cc7f04ac52f8a3bad5 243B / 243B            2.4s
=> => sha256:87b8bf94a2ace2 3.41MB / 3.41MB           3.3s
=> => extracting sha256:8a1e25ce7c4f75e372e           1.8s
=> => extracting sha256:1103112ebfc46e01c0f           0.2s
=> => extracting sha256:b4b80ef7128dc9bd114           1.0s
=> => extracting sha256:cc7f04ac52f8a3bad5b           0.0s
=> => extracting sha256:87b8bf94a2ace2b005d           0.7s
=> [internal] load build context                         0.0s
=> => transferring context: 194B                        0.0s
=> [2/4] WORKDIR /app                                  0.2s
=> [3/4] RUN pip install flask==2.3                  4.7s
=> [4/4] COPY . /app                                  0.0s
=> => exporting to image                            0.2s
=> => exporting layers                            0.2s
=> => writing image sha256:c6879156c7750c89       0.0s
=> => naming to docker.io/library/sample-flask-app:v1 0.0s
```

### What's Next?

View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

To make sure the image sample-flask-app:v1 has been successfully created, run the following command to check the list of Docker images on your system:

```
$ docker image ls
```

## **Software Engineering & Project Management Lab Experiment No: - 10**

**Aim: To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.**

The resulting output should look something like this:

\$ docker image ls					
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE	
sample-flask-app	v1	c6879156c775	10 seconds ago	147MB	
mongo	latest	24041ceefc56	6 days ago	755MB	

In the list, you should see sample-flask-app:v1, which confirms the image is now in our system. Now, run the sample-flask-app:v1 image as a container by executing the following command:

```
$ docker container run -d -p 5000:5000 sample-flask-app:v1
```

The -d flag is short for --detach and runs the container in the background. The -p flag is short for --publish and maps port 5000 of the host to port 5000 of the Docker container. After running this command, you'll see an output like this:

```
$ docker container run -d -p 5000:5000 sample-flask-app:v1  
ff37071dd4cef95cc1dc2ce7e145019339cfaec54575659f72aea4e560238f8c
```

The long string you see printed in the terminal is the container ID. To make sure the container is running, list the currently active Docker containers by running the following command:

```
$ docker container ls
```

You should see something like this:

\$ docker container ls						
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
c301c50152ac	sample-flask-app:v1	"flask run --host=0..."	14 seconds ago	Up 12 seconds	0.0.0.0:5000->5000/tcp	xenodochial_mendel

## Software Engineering & Project Management Lab Experiment No: - 10

**Aim:** To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.

**The container is up and running as expected.** Our Flask application is now running inside the container. To test it, open a web browser and go to <http://localhost:5000>. You should see the message Hello, World! displayed like this:



### SCREENSHOTS:

```
ubuntu@ip-172-31-40-218:~$ systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2023-04-10 19:46:05 UTC; 18min ago
     TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
   Main PID: 684 (dockerd)
      Tasks: 12
     Memory: 141.2M
        CPU: 3.736s
      CGroup: /system.slice/docker.service
              └─684 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Apr 10 19:46:01 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:01.899979432Z" level=info msg="ccResolverWrapper: se>
Apr 10 19:46:01 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:01.899956895Z" level=info msg="ClientConn switching >
Apr 10 19:46:02 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:02.339992511Z" level=info msg="[graphdriver] using p>
Apr 10 19:46:03 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:03.665306795Z" level=info msg="Loading containers: s>
Apr 10 19:46:04 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:04.873139021Z" level=info msg="Default bridge (docke>
Apr 10 19:46:05 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:05.081644328Z" level=info msg="Loading containers: d>
Apr 10 19:46:05 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:05.543882435Z" level=info msg="Loading containers: d>
Apr 10 19:46:05 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:05.547797680Z" level=info msg="Docker daemon" commit>
Apr 10 19:46:05 ip-172-31-40-218 systemd[1]: Started Docker Application Container Engine.
Apr 10 19:46:05 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:05.743749833Z" level=info msg="API listen on /run/do>
lines 1-22/22 (END) I

ubuntu@ip-172-31-40-218:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
ubuntu@ip-172-31-40-218:~$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu@ip-172-31-40-218:~$
```

## Software Engineering & Project Management Lab Experiment No: - 10

**Aim:** To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.

```
ubuntu@ip-172-31-40-218:~$  
ubuntu@ip-172-31-40-218:~$ pwd  
/home/ubuntu  
ubuntu@ip-172-31-40-218:~$ mkdir my-website  
ubuntu@ip-172-31-40-218:~$ cd my-website/  
ubuntu@ip-172-31-40-218:~/my-website$ wget https://www.free-css.com/assets/files/free-css-templates/download/page290/wave-cafe.zip  
--2023-04-10 20:06:14-- https://www.free-css.com/assets/files/free-css-templates/download/page290/wave-cafe.zip  
Resolving www.free-css.com (www.free-css.com)... 217.160.0.242, 2001:8d8:100f:f000::28f  
Connecting to www.free-css.com (www.free-css.com)|217.160.0.242|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 11896390 (11M) [application/zip]  
Saving to: 'wave-cafe.zip'  
  
wave-cafe.zip          100%[=====] 11.34M 6.08MB/s    in 1.9s  
  
2023-04-10 20:06:17 (6.08 MB/s) - 'wave-cafe.zip' saved [11896390/11896390]  
  
ubuntu@ip-172-31-40-218:~/my-website$  
ubuntu@ip-172-31-40-218:~/my-website$  
ubuntu@ip-172-31-40-218:~/my-website$ ls  
wave-cafe.zip  
ubuntu@ip-172-31-40-218:~/my-website$ unzip wave-cafe.zip
```

```
ubuntu@ip-172-31-40-218:~/my-website  
inflating: 2121_wave_cafe/fontawesome/webfonts/fa-regular-400.ttf  
inflating: 2121_wave_cafe/fontawesome/webfonts/fa-regular-400.woff  
inflating: 2121_wave_cafe/fontawesome/webfonts/fa-regular-400.woff2  
inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.eot  
inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.svg  
inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.ttf  
inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.woff  
inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.woff2  
creating: 2121_wave_cafe/img/  
inflating: 2121_wave_cafe/img/about-1.png  
inflating: 2121_wave_cafe/img/about-2.png  
inflating: 2121_wave_cafe/img/hot-americano.png  
inflating: 2121_wave_cafe/img/hot-cappuccino.png  
inflating: 2121_wave_cafe/img/hot-espresso.png  
inflating: 2121_wave_cafe/img/hot-latte.png  
inflating: 2121_wave_cafe/img/iced-americano.png  
inflating: 2121_wave_cafe/img/iced-cappuccino.png  
inflating: 2121_wave_cafe/img/iced-espresso.png  
inflating: 2121_wave_cafe/img/iced-latte.png  
inflating: 2121_wave_cafe/img/smoothie-1.png  
inflating: 2121_wave_cafe/img/smoothie-2.png  
inflating: 2121_wave_cafe/img/smoothie-3.png  
inflating: 2121_wave_cafe/img/smoothie-4.png  
inflating: 2121_wave_cafe/img/special-01.jpg  
inflating: 2121_wave_cafe/img/special-02.jpg  
inflating: 2121_wave_cafe/img/special-03.jpg  
inflating: 2121_wave_cafe/img/special-04.jpg  
inflating: 2121_wave_cafe/img/special-05.jpg  
inflating: 2121_wave_cafe/img/special-06.jpg  
inflating: 2121_wave_cafe/index.html  
creating: 2121_wave_cafe/js/  
inflating: 2121_wave_cafe/js/jquery-3.4.1.min.js  
creating: 2121_wave_cafe/video/  
inflating: 2121_wave_cafe/video/wave-cafe-video-bg.mp4  
ubuntu@ip-172-31-40-218:~/my-website$  
ubuntu@ip-172-31-40-218:~/my-website$  
ubuntu@ip-172-31-40-218:~/my-website$  
ubuntu@ip-172-31-40-218:~/my-website$ clear
```

## Software Engineering & Project Management Lab Experiment No: - 10

**Aim:** To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.

```
ubuntu@ip-172-31-40-218:~/my-website$ 2121_wave_cafe wave-cafe.zip
ubuntu@ip-172-31-40-218:~/my-website$ cd 2121_wave_cafe
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ ls
css fontawesome img index.html js video
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ cp -R * ../
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ rm -rf wave-cafe.zip 2121_wave_cafe
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ cd ..
ubuntu@ip-172-31-40-218:~/my-website$ ls
css fontawesome img index.html js video
ubuntu@ip-172-31-40-218:~/my-website$ nano Dockerfile
```

```
GNU nano 6.2                                     Dockerfile
FROM httpd:2.4
COPY . /usr/local/apache2/htdocs/
```

## Software Engineering & Project Management Lab Experiment No: - 10

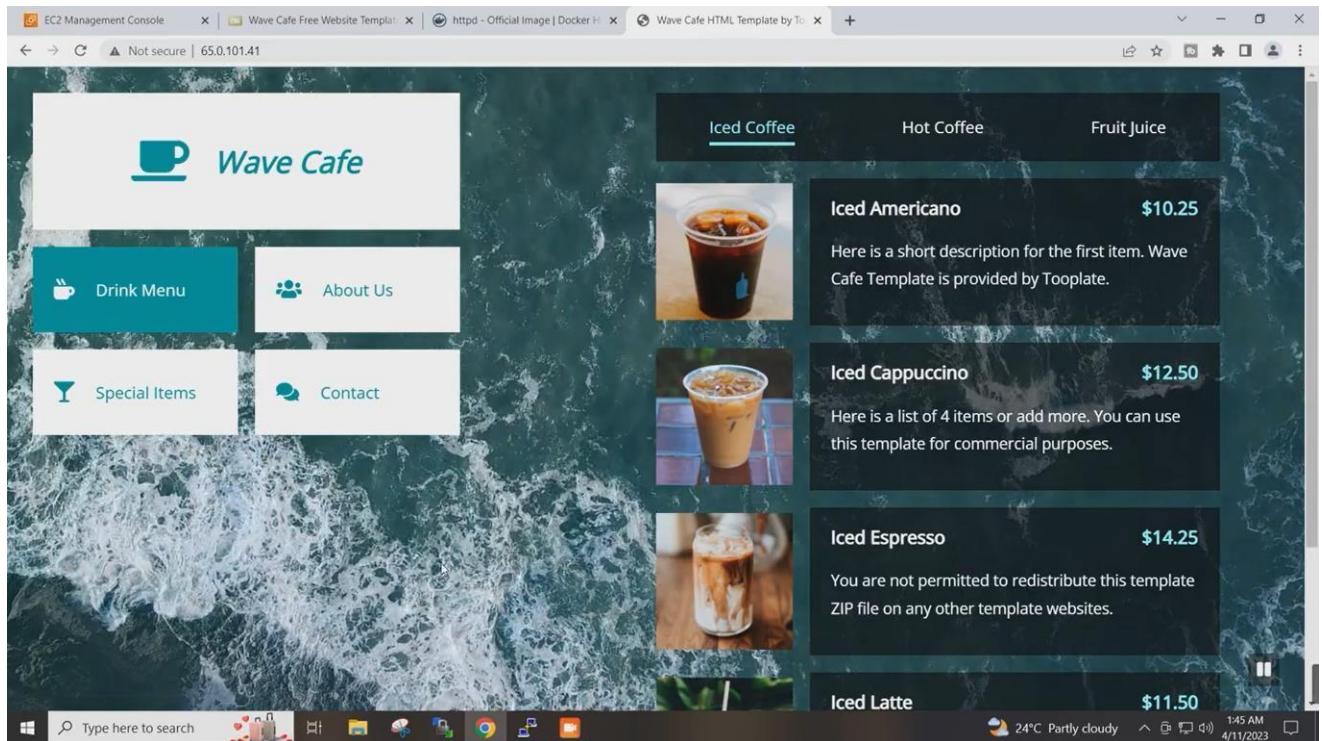
**Aim:** To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.

```
ubuntu@ip-172-31-40-218:~/my-website$ ls
ubuntu@ip-172-31-40-218:~/my-website$ ls
2121_wave_cafe wave-cafe.zip
ubuntu@ip-172-31-40-218:~/my-website$ cd 2121_wave_cafe
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ ls
css fontawesome img index.html js video
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ cp -R * ../.
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ 
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ 
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ cd ..
ubuntu@ip-172-31-40-218:~/my-website$ ls
2121_wave_cafe css fontawesome img index.html js video wave-cafe.zip
ubuntu@ip-172-31-40-218:~/my-website$ rm -rf wave-cafe.zip 2121_wave_cafe
ubuntu@ip-172-31-40-218:~/my-website$ 
ubuntu@ip-172-31-40-218:~/my-website$ ls
css fontawesome img index.html js video
ubuntu@ip-172-31-40-218:~/my-website$ nano Dockerfile
ubuntu@ip-172-31-40-218:~/my-website$ ls
Dockerfile css fontawesome img index.html js video
ubuntu@ip-172-31-40-218:~/my-website$ docker build . -t my-website:latest
Sending build context to Docker daemon 13.61MB
Step 1/2 : FROM httpd:2.4
2.4: Pulling from library/httpd
f1f26f570256: Pull complete
a6b093ae1967: Pull complete
6b400bbb27df: Pull complete
6e310dd059b6: Pull complete
471cb5914961: Pull complete
Digest: sha256:4055b18d92fd006f74d4a2aac172a371dc9a750eaa78000756dee55a9beb4625
Status: Downloaded newer image for httpd:2.4
--> dcl1a95e13784
Step 2/2 : COPY . /usr/local/apache2/htdocs/
--> 7d48427f5e2f
Successfully built 7d48427f5e2f
Successfully tagged my-website:latest
ubuntu@ip-172-31-40-218:~/my-website$ 
ubuntu@ip-172-31-40-218:~/my-website$ 
ubuntu@ip-172-31-40-218:~/my-website$ cleaR
```

```
ubuntu@ip-172-31-40-218:~/my-website$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
my-website latest 7d48427f5e2f 15 seconds ago 159MB
httpd 2.4 dcl1a95e13784 4 days ago 145MB
ubuntu@ip-172-31-40-218:~/my-website$ docker run -d -p 80:80 my-website:latest
e0a6df73ab6718a1b648d9b5f00dcc89e846d1fe12bd568ce9b1412fc0d3c9da
ubuntu@ip-172-31-40-218:~/my-website$ 
ubuntu@ip-172-31-40-218:~/my-website$ 
ubuntu@ip-172-31-40-218:~/my-website$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
NAMES
e0a6df73ab67 my-website:latest "httpd-foreground" 8 seconds ago Up 7 seconds 0.0.0.0:80->80/tcp, :::80->80/tcp
trusting_rosalind
ubuntu@ip-172-31-40-218:~/my-website$ 
```

## Software Engineering & Project Management Lab Experiment No: - 10

**Aim:** To learn Dockerfile instructions, build an image for a sample web application using DOCKERFILE.



**Conclusion:** Thus, we have successfully learnt Dockerfile instructions & build an image for a sample web application using DOCKERFILE.