# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**

**Aim:** 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

## Theory:

Continuous Integration (CI) is a software development practice where developers frequently integrate their code changes into a shared repository—often multiple times per day. Each integration is verified through an automated build and test process to detect errors as early as possible.

**Jenkins** is an open-source automation server used to implement CI/CD pipelines. It supports building, testing, and deploying applications through customizable jobs. Jenkins can execute scripts, compile code, run tests, and more, using plugins and freestyle or pipeline projects.

### Key Features of Jenkins:

* Open-source and extensible
* Support for various plugins **(Maven, Git, Docker, etc.)**
* Freestyle and pipeline job support
* Easy integration with Git, GitHub, Bitbucket
* Parameterised builds and scheduling
* Build monitoring through Console Output and Blue Ocean UI **Demonstration of Jenkins Job Execution (Theoretical Steps):**  Example 1.1: Deploying a Freestyle App in Jenkins

1. Login to Jenkins Dashboard.
2. Create a new item → Name it → Select **Freestyle project**.
3. Go to **Build Section** → Select **Execute Shell**.

*4.* Enter shell command, e.g.:

*echo "Hello, Jenkins!"*

1. Click **Apply** and **Save**.
2. Click **Build Now**.
3. View **Console Output** to verify execution.

Example 2.1: Running a Script with Parameters

1. Create a shell script (e.g., example1.cmd):

*echo "Welcome, $1!"*

1. Test on terminal:

*sh example1.cmd John*

1. Modify Jenkins job to run the script with parameters:

*sh example1.cmd John*

1. Build and check console output for the message.

TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**

*Running a Java Program under Jenkins*

1. Write a simple Java program:

*public class Hello {*

*public static void main(String[] args) { System.out.println("Hello from Java!");*

*}*

*}*

1. Compile and run in terminal:

*javac Hello.java java Hello*

*3.* Create a Jenkins freestyle project → **Execute Shell**:

*javac Hello.java java Hello*

4. Build and check the output.

Example 3.1: Parameterised Build

1. Create a new freestyle project.
2. Enable **"This project is parameterized"**.
3. Add a **String parameter** (e.g., Fname).
4. Add a **Choice parameter** (e.g., City) with values: Mumbai, Pune, Delhi.

### In Build → Execute Shell:

*echo "Hello $Fname from $City!"*

1. Click **Build with Parameters** and enter values.
2. Output displays personalized greeting.

Example 4.1: Running a Python Program

1. Write a Python script (greet.py):

*import sys*

*print(f"Hello, {sys.argv[1]}!")*

1. Run in terminal:

*python3 greet.py Alice*

TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**

1. Create Jenkins freestyle project.
2. Enable parameterization → Add **String parameter**: Name
3. Configure build step:

*python3 greet.py $Name*

1. Build with parameter Name = Alice.
2. Output: Hello, Alice!

### Use Case Examples:

* Automating builds and deployments
* Testing code on every commit
* Running scripts or data processing pipelines
* Hosting Java or Python apps using Tomcat
* Scheduled jobs for cleanup, backup, etc.

**Implementation:**

### 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:



Naming the job and setting it as freestyle:

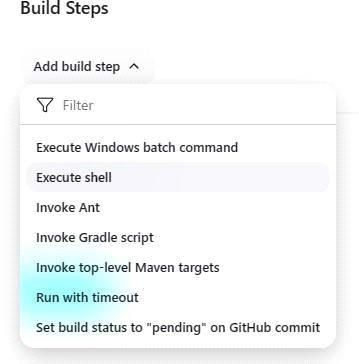
TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**



Selecting build type as “Execute shell”:

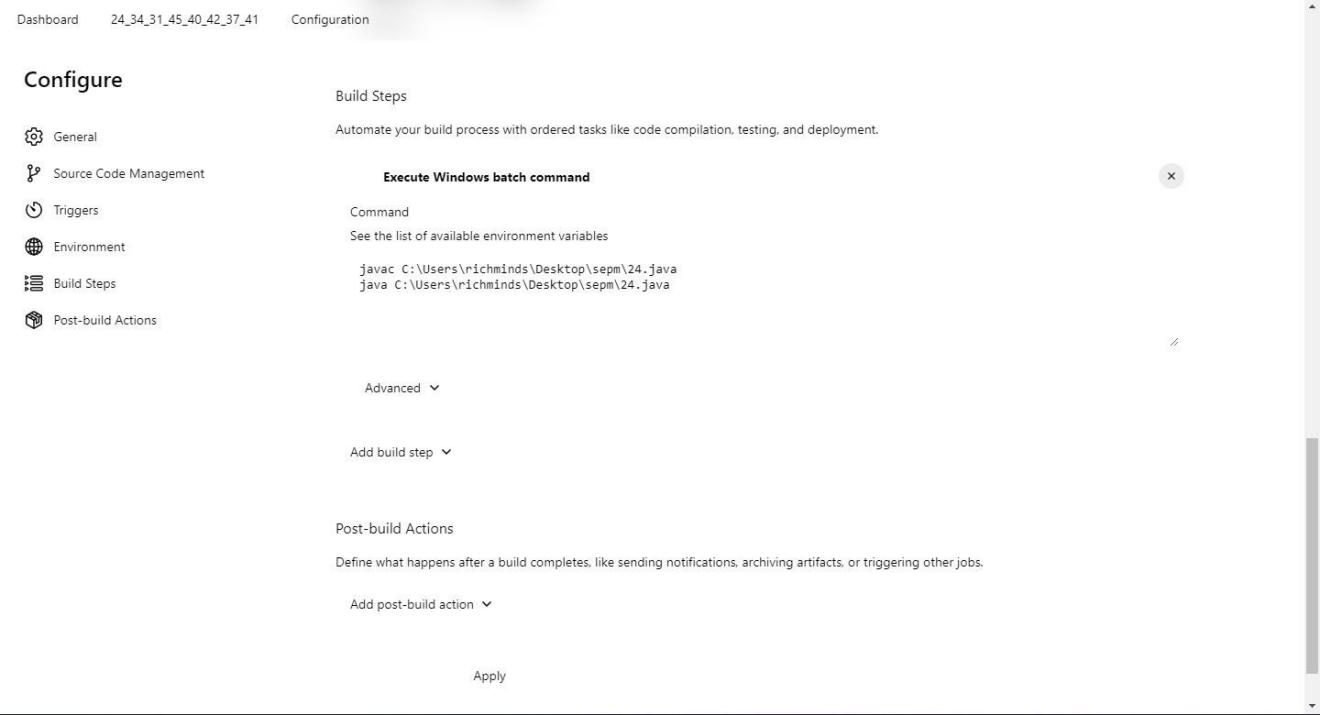


Entering a simple command for the shell execution:

TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**



Applying and saving the project configuration:



Building the project:



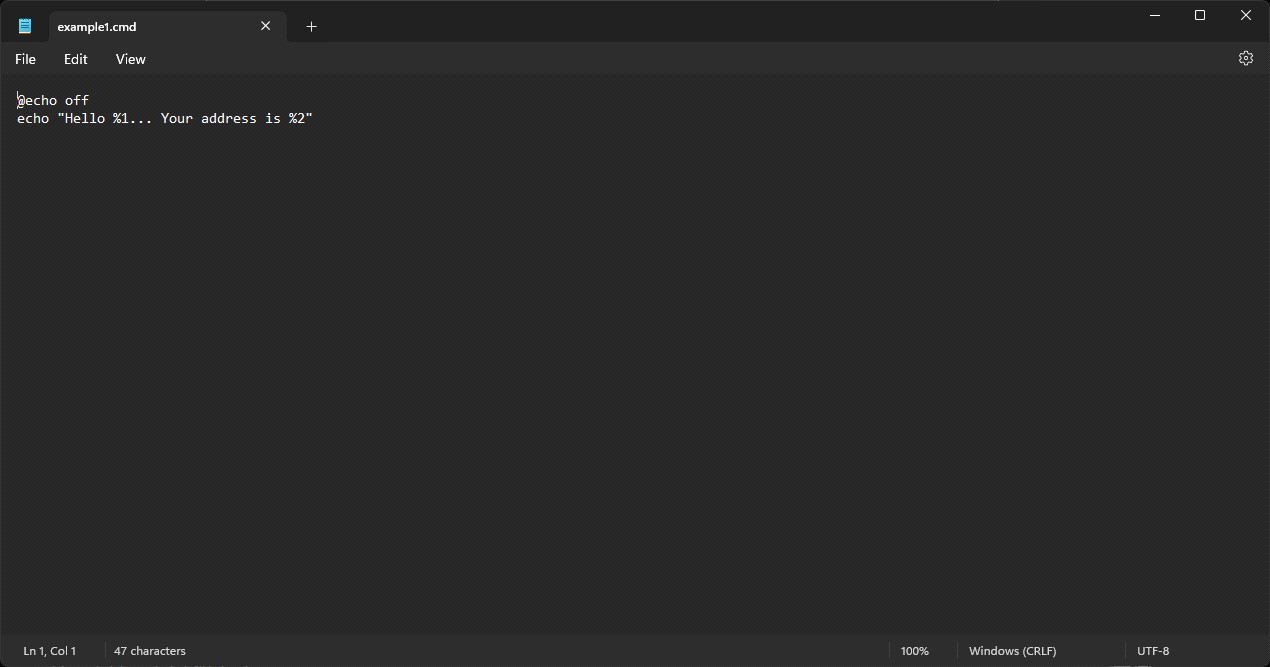
Console output (after building):



TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

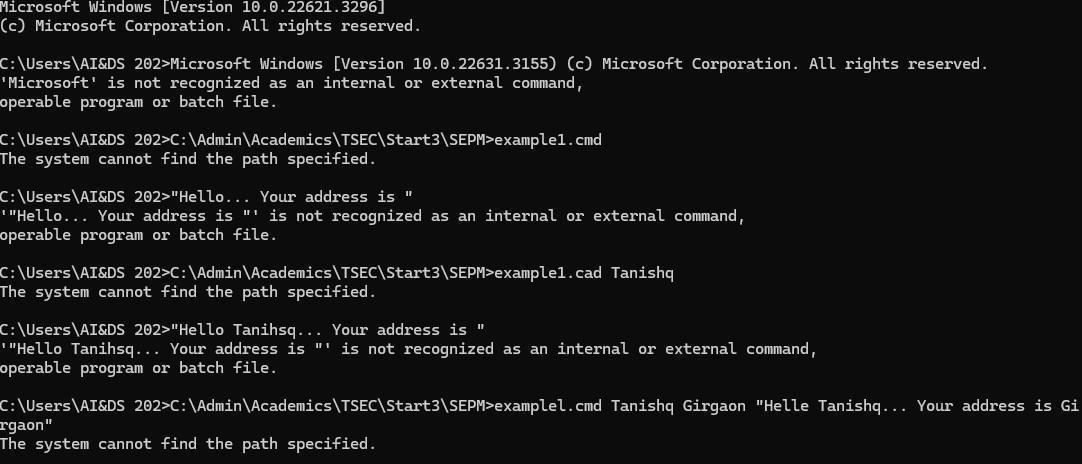
**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**



Example 1.2: Taking parameters through files Contents of script

example1.cmd:

Executing script example1.cmd on the terminal:



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



Console output after building the modified project:

TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

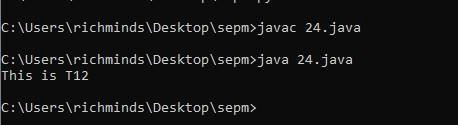
# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**

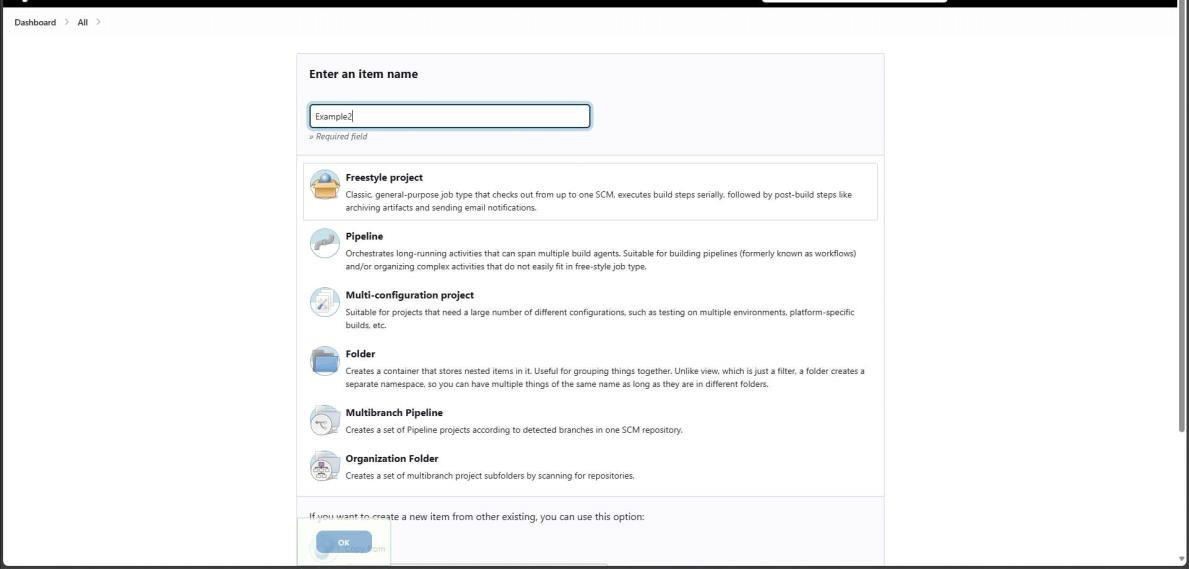


Creating a simple Java program:

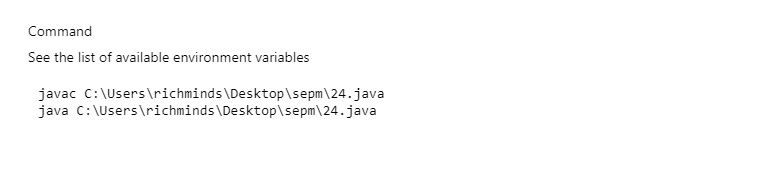
Compiling and running the program on the terminal:



Creating a new freestyle project:



Configure new project:



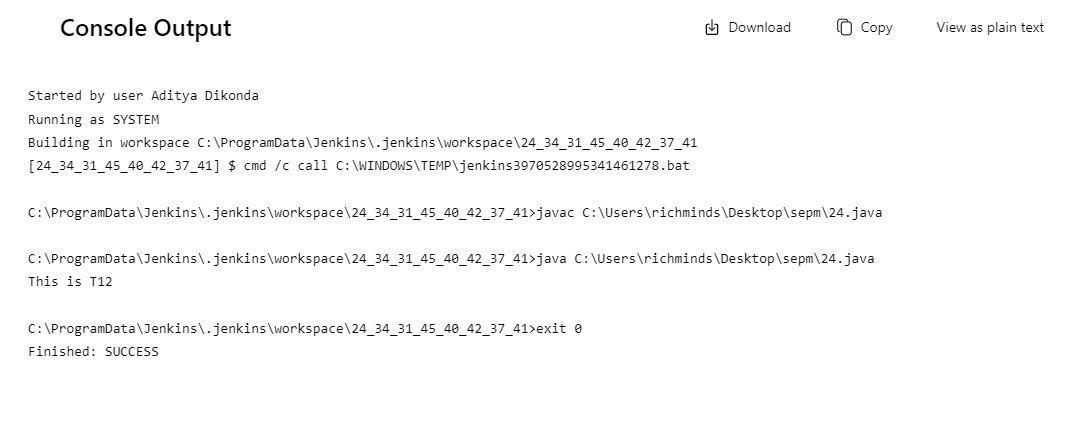
Console output after building:

: Running a Java program under Jenkins

TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**

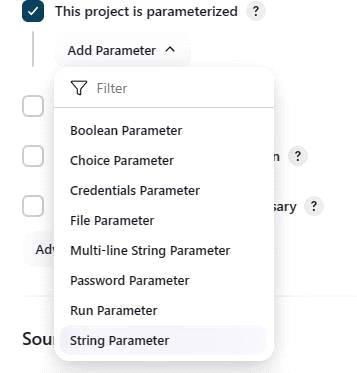


Example 3.1: Parameterise build Creating a new

Example 3

freestyle project:

Enabling parameterisation and adding a String parameter:

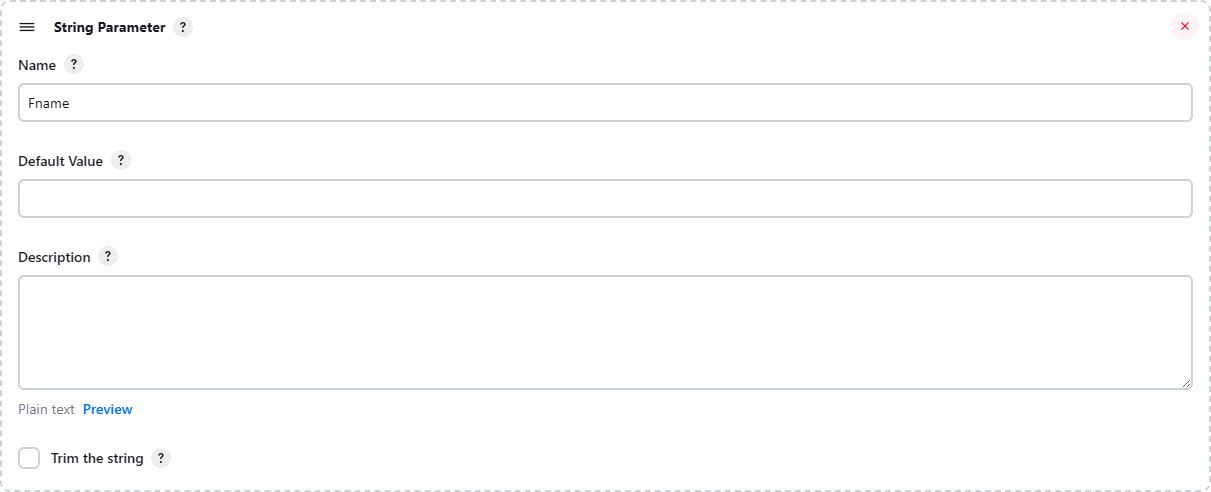


Configuring the string parameter as Fname:

TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

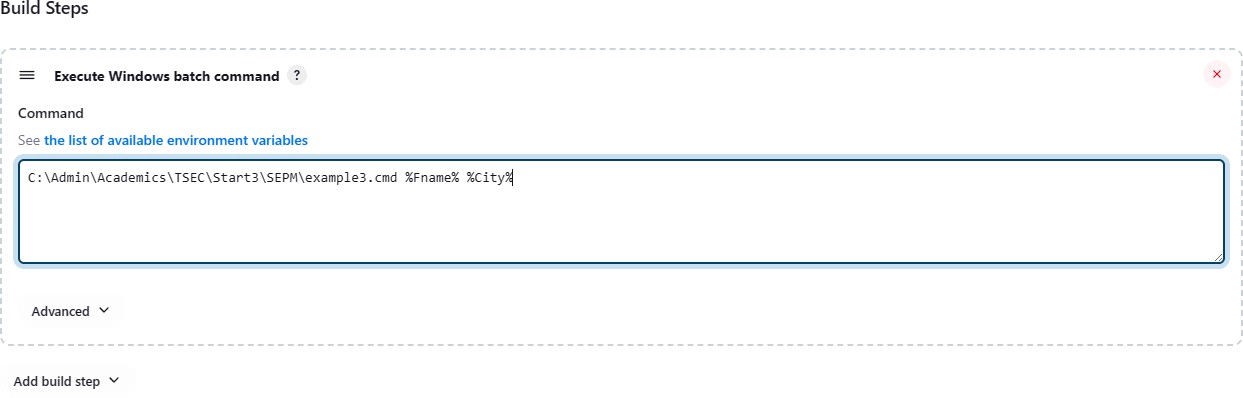
**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**



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



Configuring build steps:



Entering parameters for build:



TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

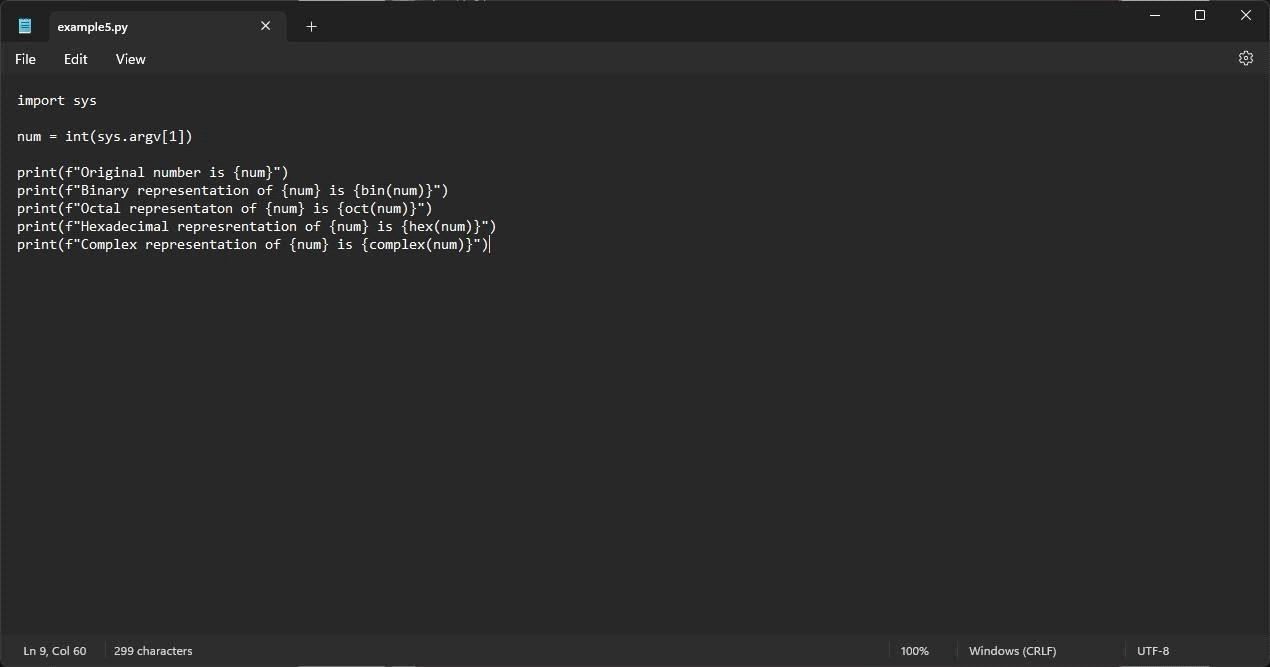
# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**

Console output after building:



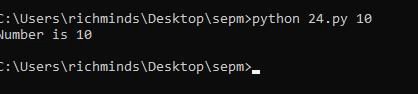
Example 5



Example 5.1: Running a Python program Creating a simple

Python script:

Running the Python script on the terminal:

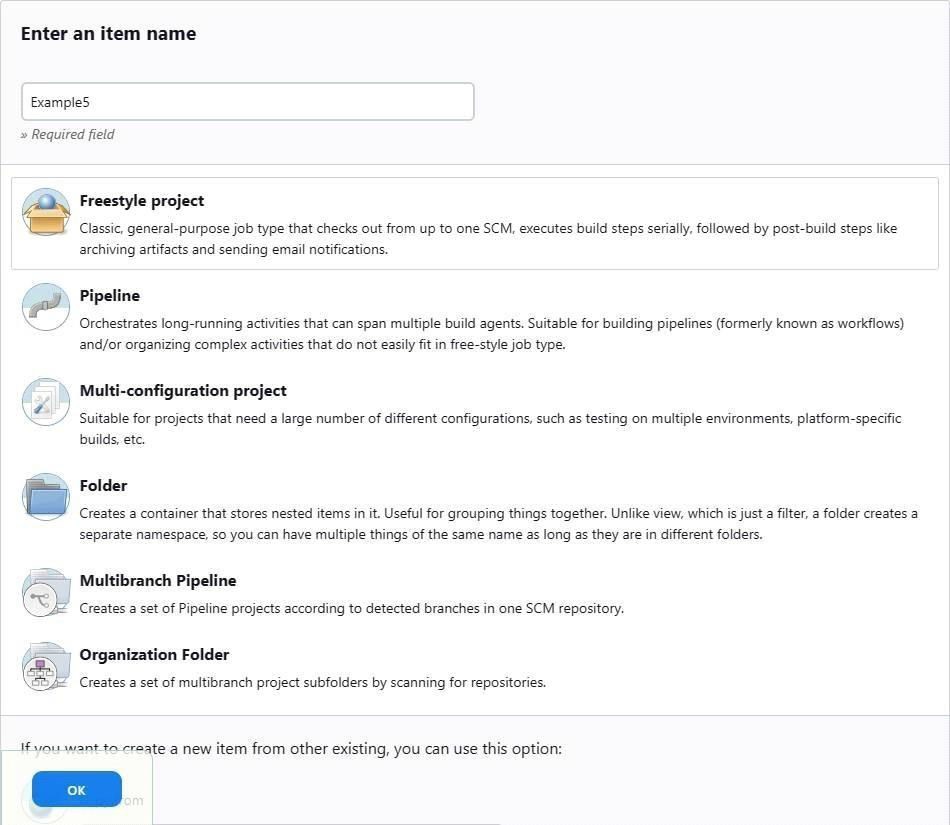


Creating a new freestyle project:

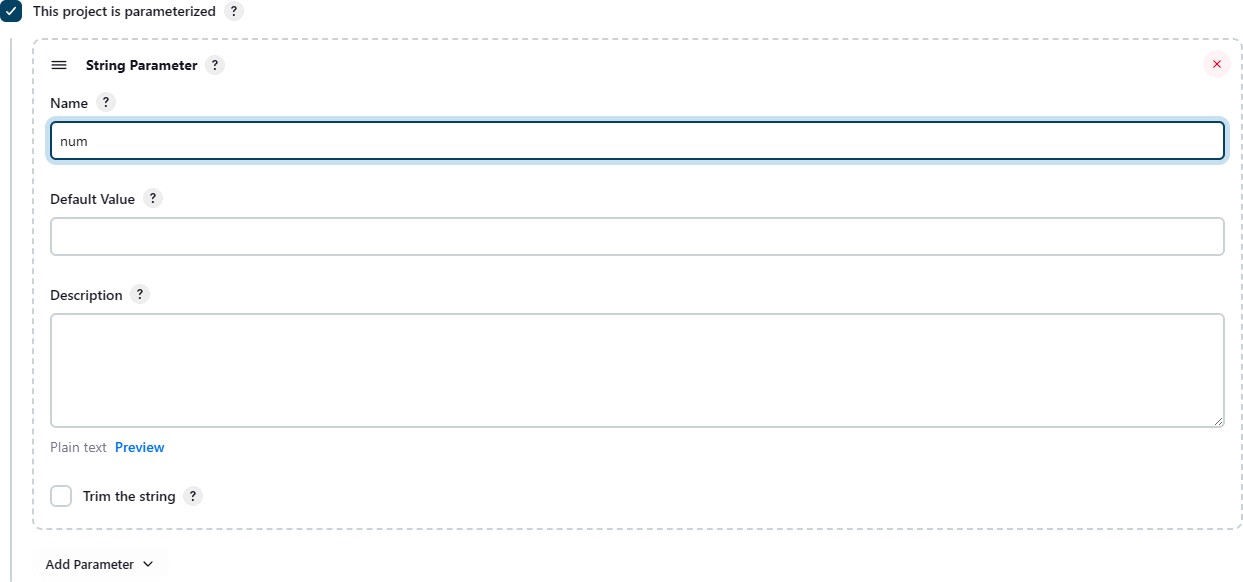
TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**



Parameterising the project with a string parameter as follows:

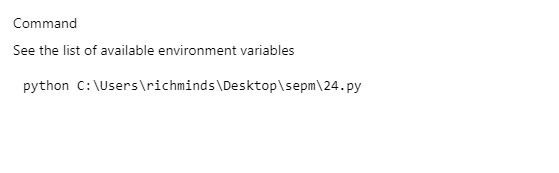


Configuring the build steps:

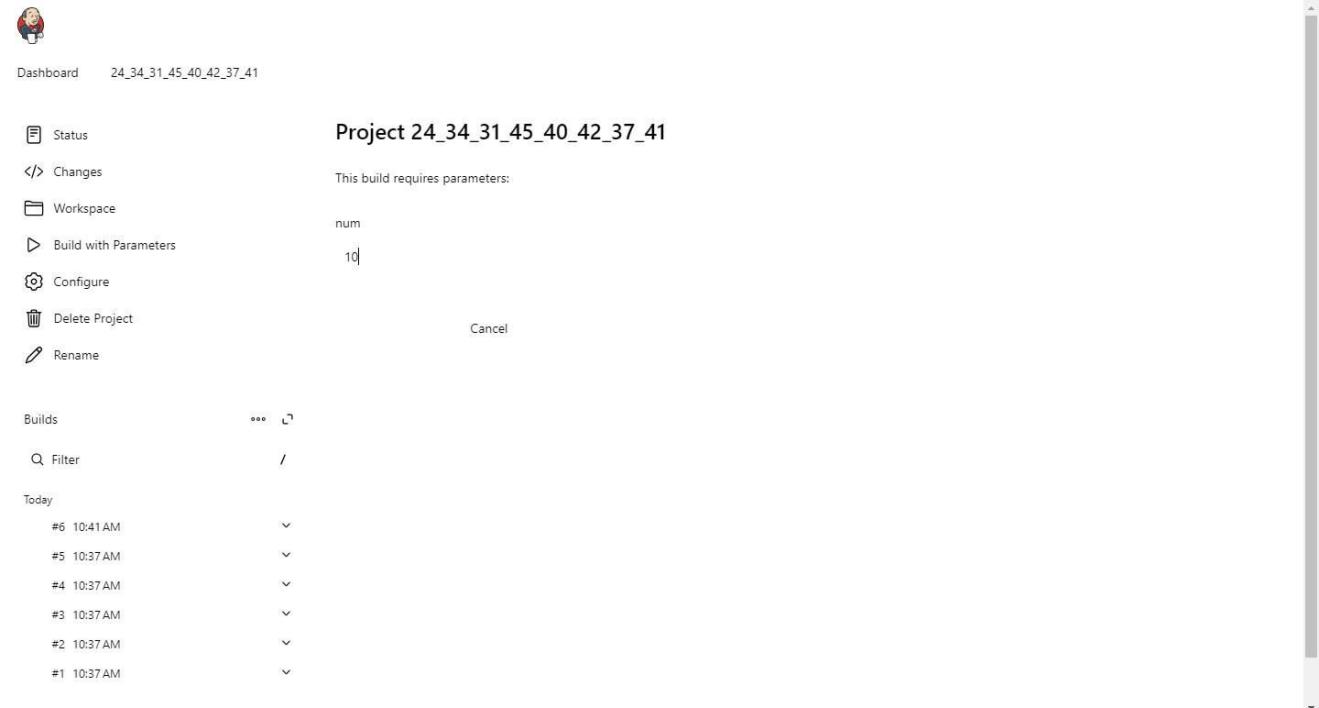
TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35

# Software and Project Management Lab Experiment No: - 05

**Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins**



Setting the parameter for the build:



**Conclusion:** We have successfully built the pipeline of jobs using Maven / Gradle / Ant in Jenkins, created a pipeline script to Test and deploy an application over the tomcat server.

**LO Mapping:** *LO is mapped*

TSEC Batch:-T12 Name & Roll No:- Sarthak Hinge - 35