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!"
5. Click **Apply** and **Save**.
6. Click **Build Now**.
7. 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!"
2. Test on terminal:
   sh example1.cmd John
3. Modify Jenkins job to run the script with parameters:

```
sh example1.cmd John
```

4. Build and check console output for the message.

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!");
   }
}
```

2. 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.
5. In **Build → Execute Shell**: *echo "Hello $Fname from $City!"*

6. Click **Build with Parameters** and enter values.
7. 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]}!")
```

2. Run in terminal:

python3 greet.py Alice

3. Create Jenkins freestyle project.
4. Enable parameterization → Add **String parameter**: Name 5. Configure build step:

   python3 greet.py $Name

6. Build with parameter Name = Alice.
7. 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:

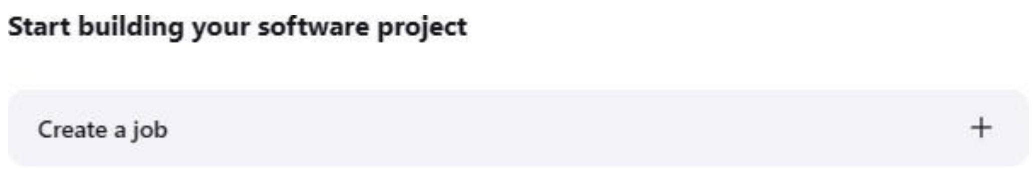
Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as 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.

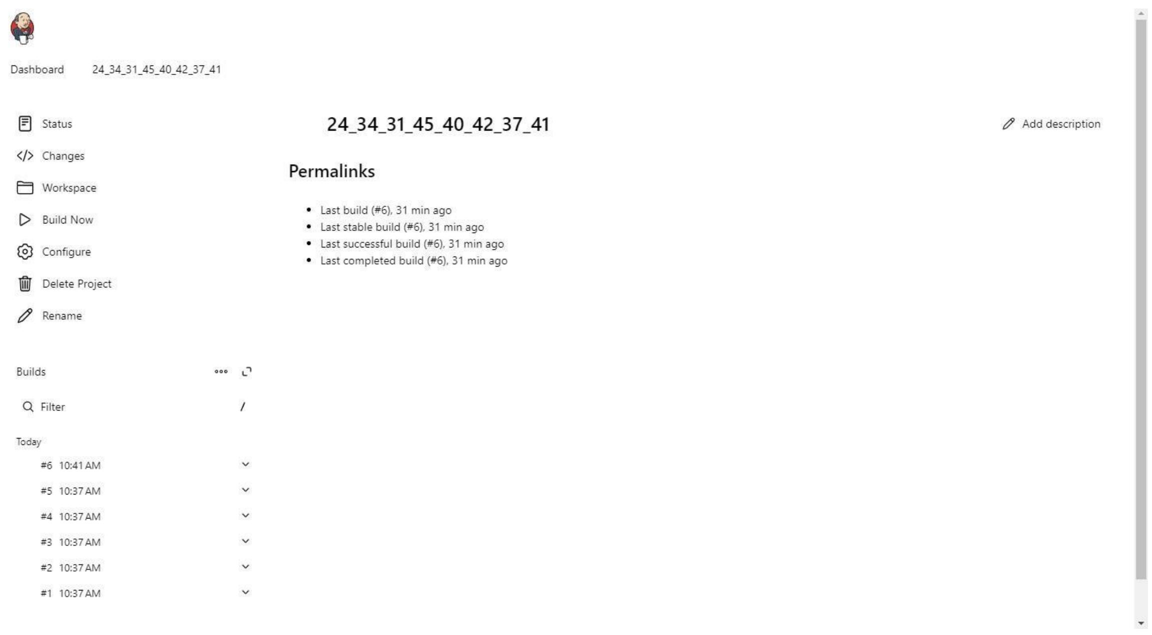
Example 1

Example 1.1: Deploying a freestyle app in Jenkins Creating a job:



Naming the job and setting it as freestyle:

Software and Project Management Lab Experiment No: - 05 Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins



Dashboard 24_34_31_45_40_42_37_41

24_34_31_45_40_42_37_41

Add description

Status
Changes
Workspace
Build Now
Configure
Delete Project
Rename

Permalinks

- Last build (#6), 31 min ago
- Last stable build (#6), 31 min ago
- Last successful build (#6), 31 min ago
- Last completed build (#6), 31 min ago

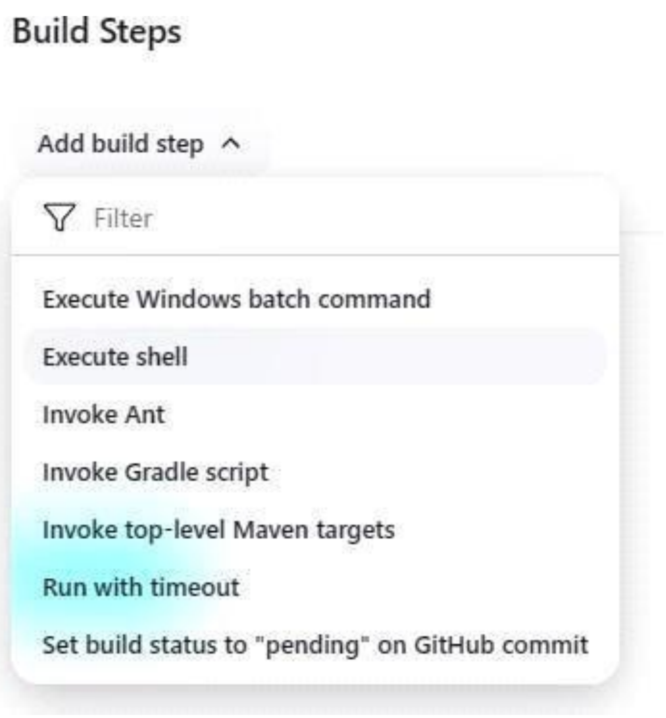
Builds

Filter

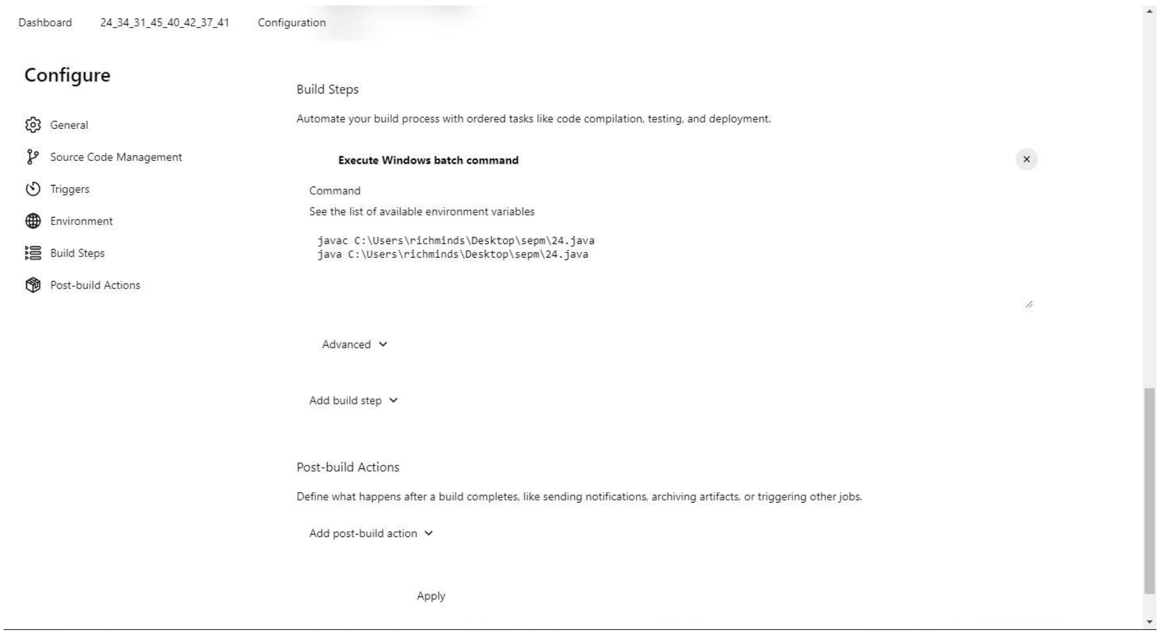
Today

#6 10:41 AM
#5 10:37 AM
#4 10:37 AM
#3 10:37 AM
#2 10:37 AM
#1 10:37 AM

Selecting build type as “Execute shell”:


Build Steps
Add build step
Filter
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit

Entering a simple command for the shell execution:



Build Steps

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

Execute Windows batch command

Command

See the list of available environment variables

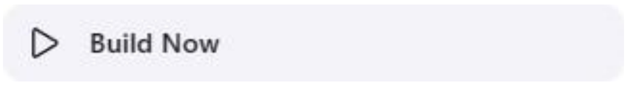
```
javac C:\Users\richminds\Desktop\sepm\24.java
java C:\Users\richminds\Desktop\sepm\24.java
```

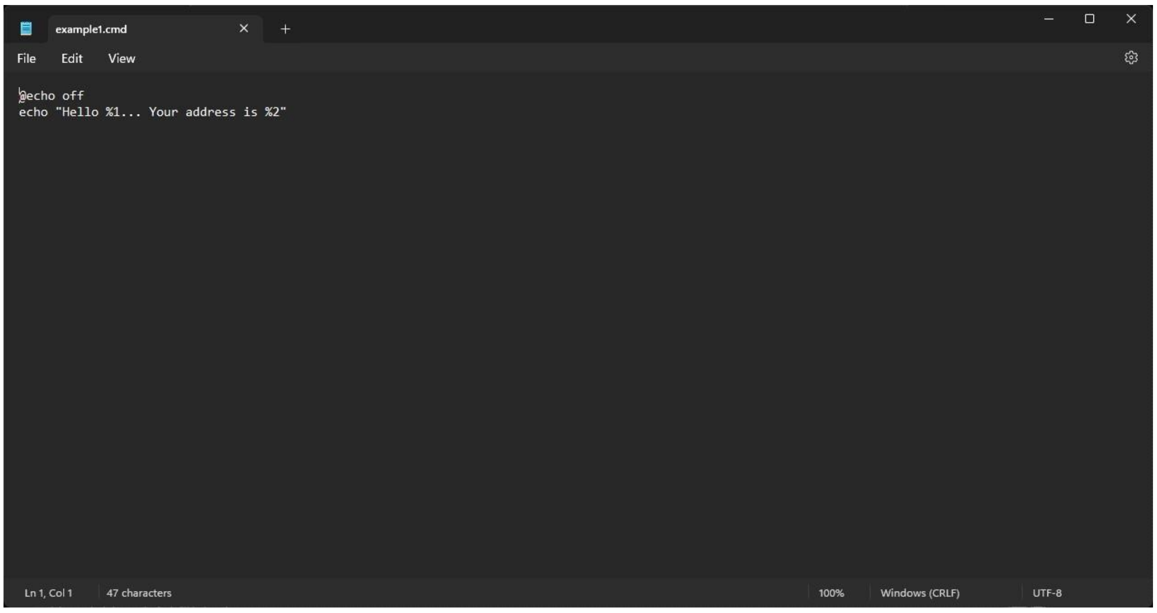
Post-build Actions

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

Applying and saving the project configuration:

Building the project:





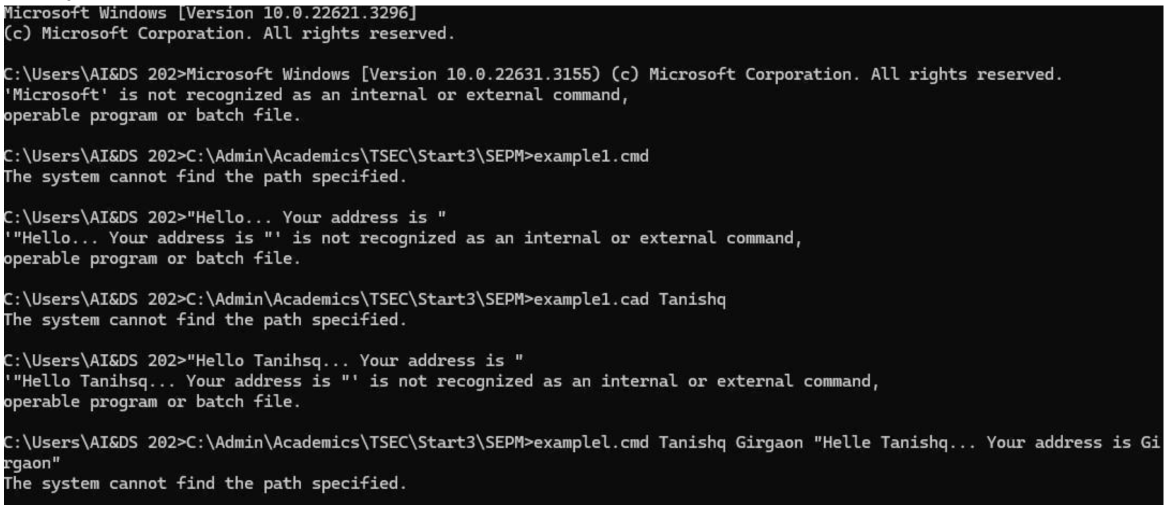
```
@echo off
echo "Hello %1... Your address is %2"
```

Software and Project Management Lab Experiment No: - 05 Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins

<u>Example 1.2: Taking parameters through files</u> Contents of script

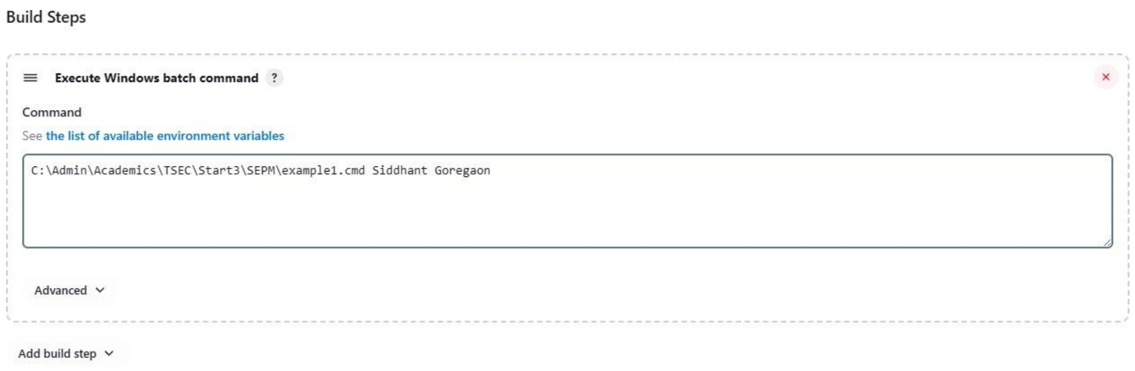
`example1.cmd`:

Executing script `example1.cmd` on the terminal:



```
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 Tanihsq... Your address is "
'"Hello Tanihsq... 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>examplel.cmd Tanishq Girgaon "Helle Tanishq... Your address is Gi
rgaon"
The system cannot find the path specified.
```

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



Build Steps

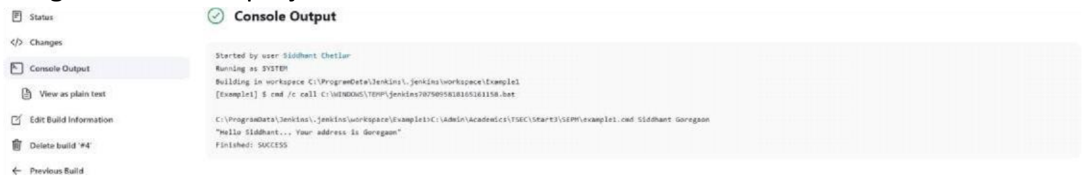
Execute Windows batch command

Command

See the list of available environment variables

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

Console output after building the modified project:

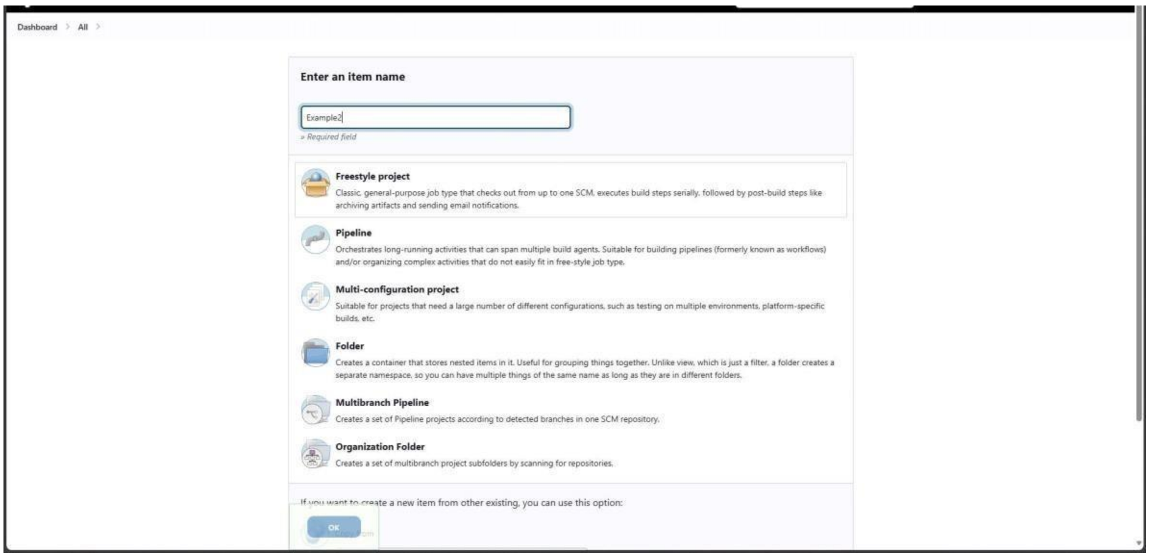


Console Output

```
Started by user Siddhant Chetlur
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\Example1
[Example1] $ cmd /c call C:\WINDOWS\TEMP\jenkins7075095818165161158.bat

C:\ProgramData\Jenkins\.jenkins\workspace\Example1>C:\Admin\Academics\TSEC\Start3\SEPM\example1.cmd Siddhant Goregaon
"Hello Siddhant... Your address is Goregaon"
Finished: SUCCESS
```

Creating a new freestyle project:

Software and Project Management Lab Experiment No: - 05 Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins



Dashboard > All >

Enter an item name

Example2

» 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.

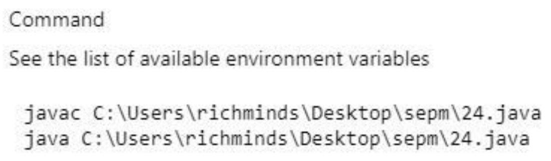
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

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:

Configure new project:

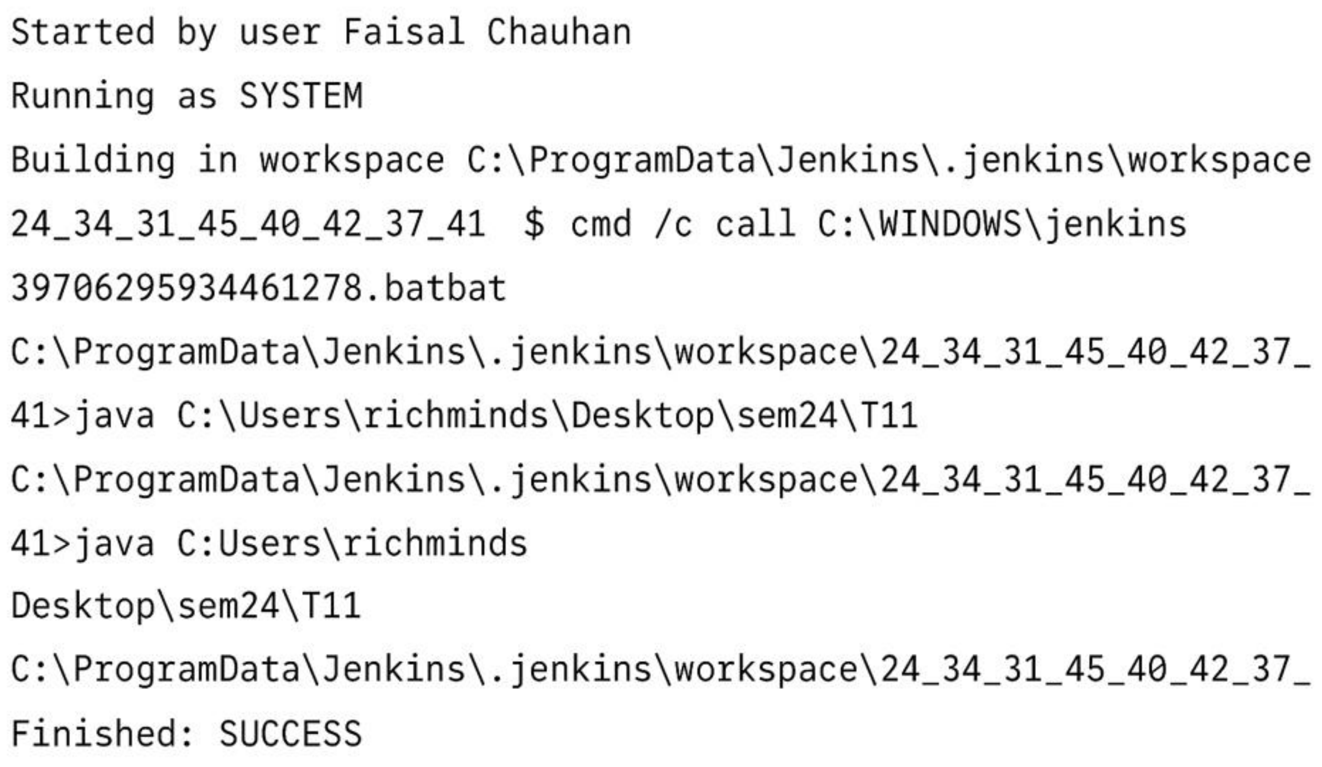


Command

See the list of available environment variables

```
javac C:\Users\richminds\Desktop\sepm\24.java
java C:\Users\richminds\Desktop\sepm\24.java
```

Console output after building:

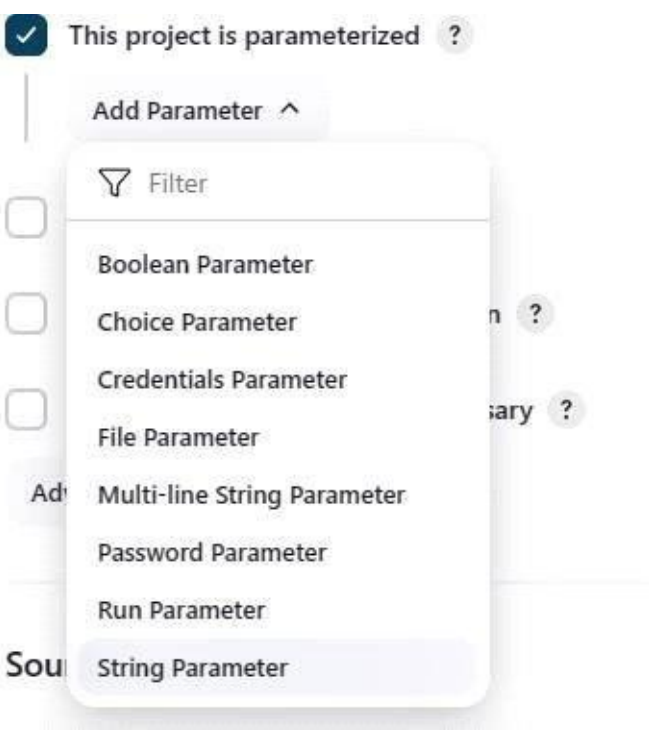


```
Started by user Faisal Chauhan
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace
24_34_31_45_40_42_37_41  $ cmd /c call C:\WINDOWS\jenkins
3970629593446l278.batbat
C:\ProgramData\Jenkins\.jenkins\workspace\24_34_31_45_40_42_37_
41>java C:\Users\richminds\Desktop\sem24\T11
C:\ProgramData\Jenkins\.jenkins\workspace\24_34_31_45_40_42_37_
41>java C:Users\richminds
Desktop\sem24\T11
C:\ProgramData\Jenkins\.jenkins\workspace\24_34_31_45_40_42_37_
Finished: SUCCESS
```

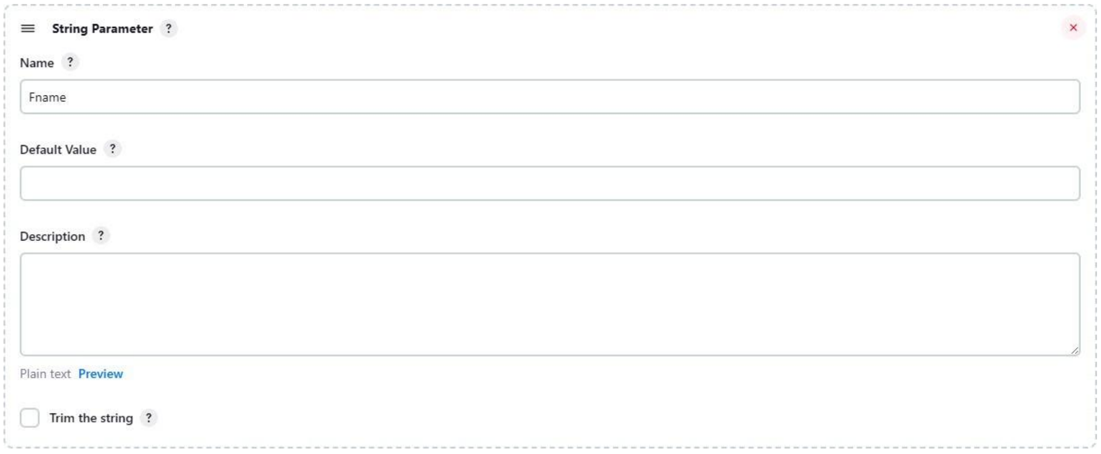
Example 3

Example 3.1: Parameterise build Creating a new freestyle project:

Enabling parameterisation and adding a String parameter:



Configuring the string parameter as `Fname`:



String Parameter

Name: Fname

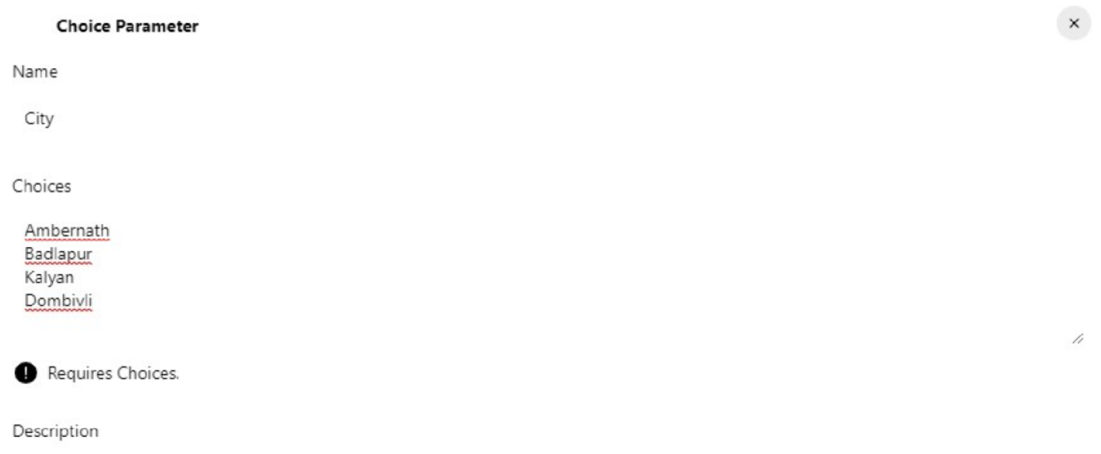
Default Value:

Description:

Plain text Preview

Trim the string

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



Choice Parameter

Name: City

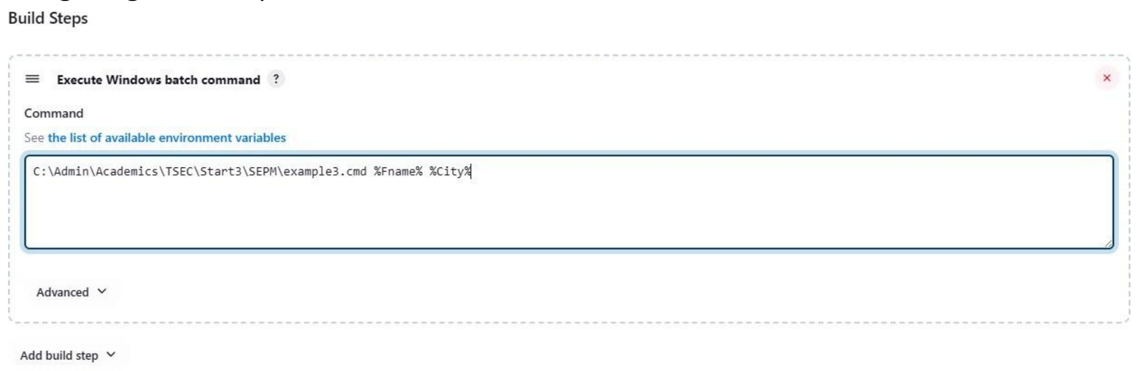
Choices:
Ambernath
Badlapur
Kalyan
Dombivli

Requires Choices.

Description

Software and Project Management Lab Experiment No: - 05 Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins

Configuring build steps:



Build Steps

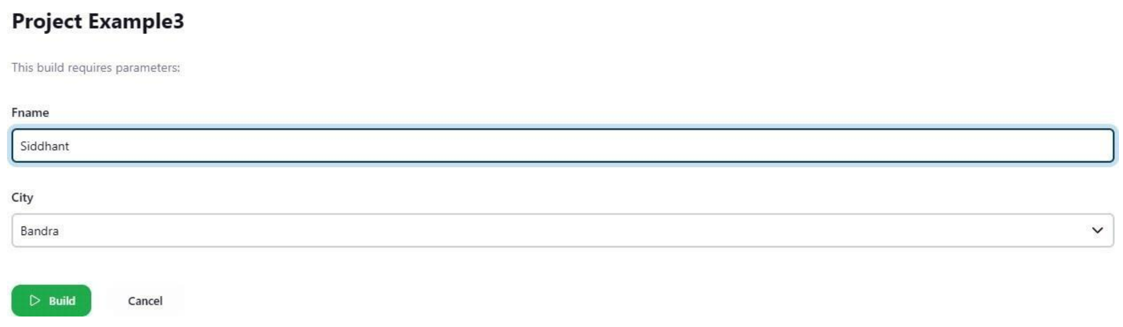
Execute Windows batch command

Command

See the list of available environment variables

```
C:\Admin\Academics\TSEC\Start3\SEPM\example3.cmd %Fname% %City%
```

Entering parameters for build:



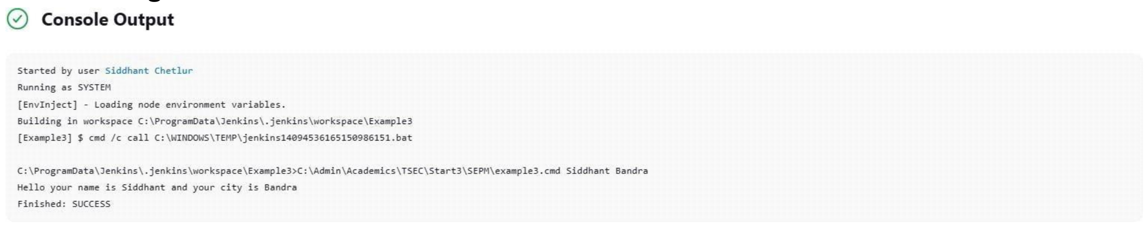
Project Example3

This build requires parameters:

Fname: Siddhant

City: Bandra

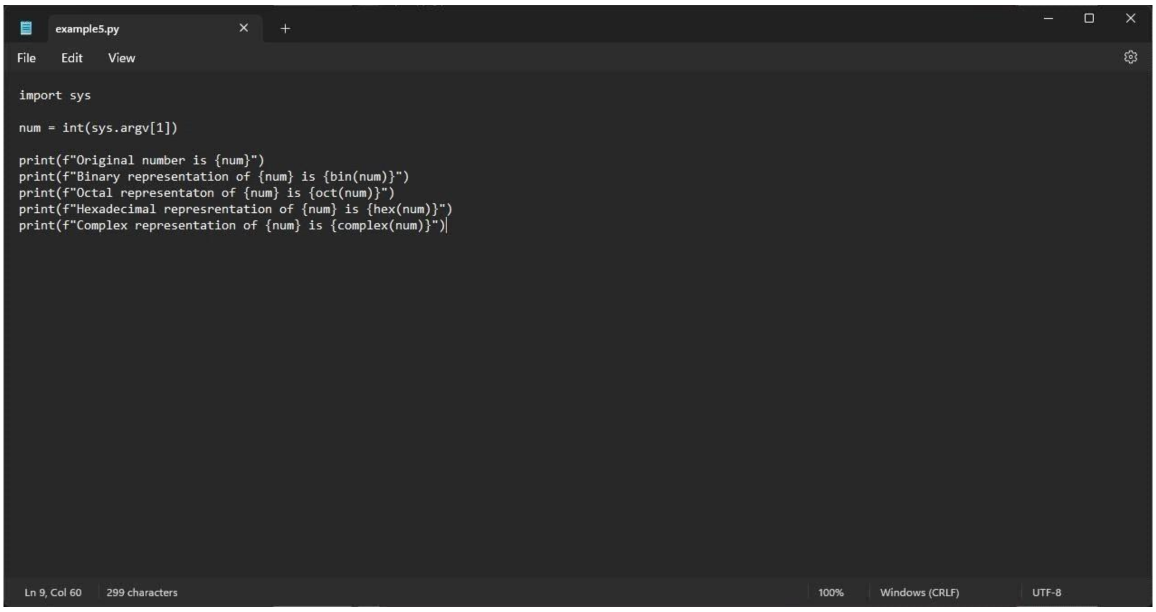
Console output after building:



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
```

<u>Example 5</u>

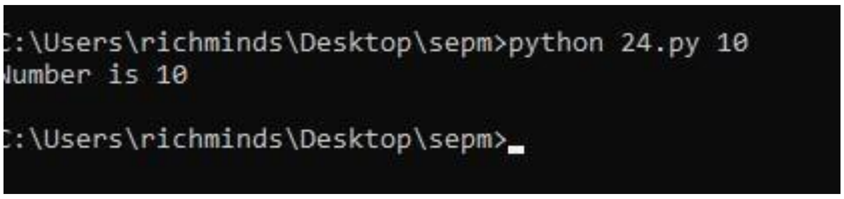


```
import sys

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 represrentation of {num} is {hex(num)}")
print(f"Complex representation of {num} is {complex(num)}")
```

<u>Example 5.1: Running a Python program</u> Creating a simple Python

script:

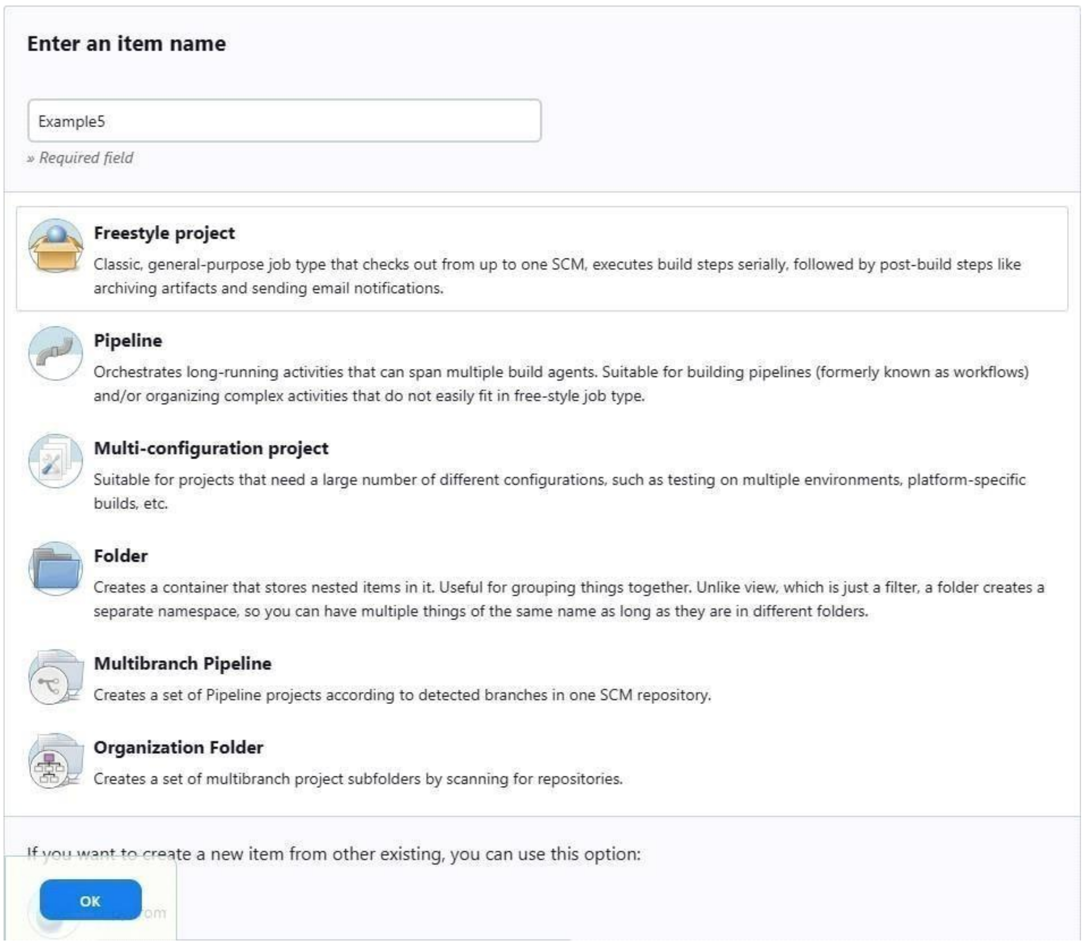
Running the Python script on the terminal:



```
:\Users\richminds\Desktop\sepm>python 24.py 10
Number is 10

:\Users\richminds\Desktop\sepm>
```

Creating a new freestyle project:

Software and Project Management Lab Experiment No: - 05 Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins



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.

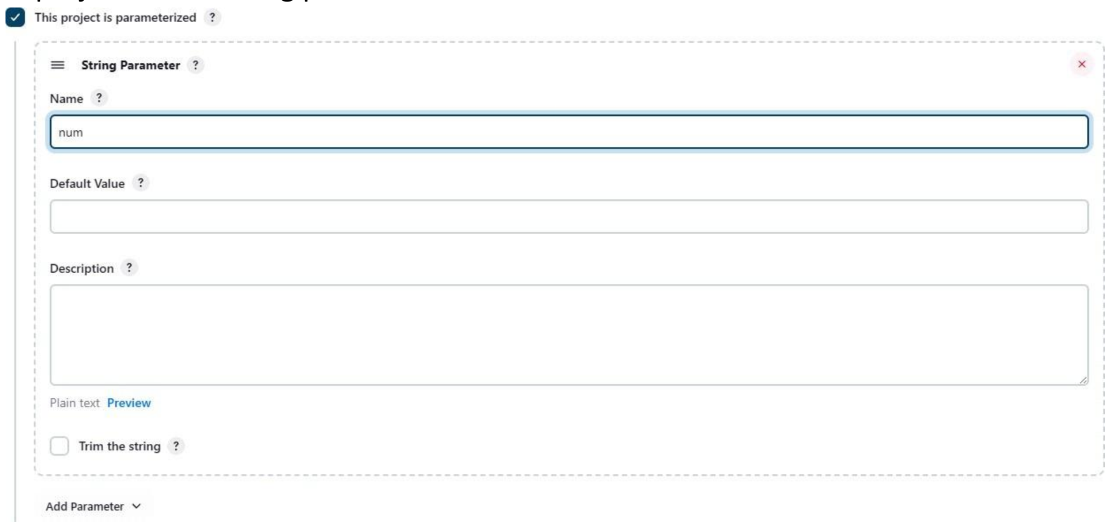
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

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:

Parameterising the project with a string parameter as follows:



☑ This project is parameterized

String Parameter

Name: num

Default Value:

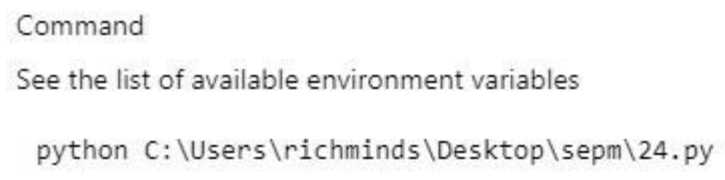
Description:

Plain text Preview

☐ Trim the string

Configuring the build steps:

Software and Project Management Lab Experiment No: - 05 Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins

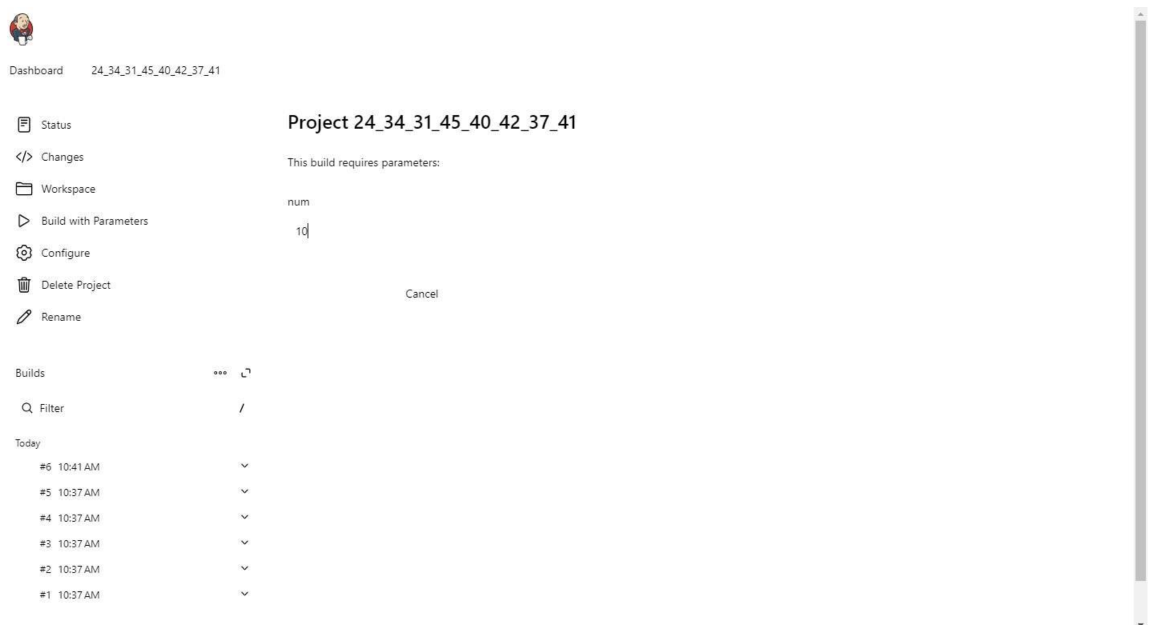


Command

See the list of available environment variables

```
python C:\Users\richminds\Desktop\sepm\24.py
```

Setting the parameter for the build:



Project 24_34_31_45_40_42_37_41

This build requires parameters:

num

10

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*

T11