Experiment 10:

To learn Software Configuration Management and provisioning using Puppet Blocks(Manifest, Modules, Classes, Function)

Puppet manifests are configuration files written in Puppet's declarative language that define the desired state of system resources like packages, files, and services. They are the core building blocks of Puppet automation.

Step	Purpose	Outcome
Install Puppet Agent	Enables manifest execution	Installed via Chocolatey
Create Manifest Folder	Organizes configuration files	`C:\puppet` created
Write Manifest	Defines Git installation	Needs Chocolatey provider
Apply Manifest	Executes configuration	Fails without provider
Install Chocolatey	Adds Windows package manager	Required for Puppet on Windows
Install Puppet Module	Integrates Chocolatey with Puppet	Adds Chocolatey provider
Final Manifest	Automates setup and provisioning	Ready for execution

PS C:\Users\tejah> choco install puppet-agent

Do you want to continue?([Y]es/[N]o): y

PS C:\Users\tejah> New-Item -Path "C:\puppet" -ItemType Directory -Force

Directory: C:

Mode	LastWriteTime	Length	Name
d-----	07-10-2025 21:37		puppet

```
PS C:\Users\tejah> Set-Content -Path "C:\puppet\site.pp" -Value @"
>> class install_git {
>>   package { 'git':
>>     ensure => installed,
>>   }
>> }
>>
>> include install_git
>> "@
PS C:\Users\tejah> Get-Content "C:\puppet\site.pp"
class install_git {
  package { 'git':
    ensure => installed,
  }
}

include install_git

PS C:\Users\tejah> puppet apply C:\puppet\site.pp
Notice: Compiled catalog for laptop-qup1o2oi in environment production in 3.65 seconds
Error: The source parameter is required when using the Windows provider.
Error: /Stage[main]/Install_git/Package[git]/ensure: change from 'absent' to 'present' failed:
The source parameter is required when using the Windows provider.
Notice: Applied catalog in 0.64 seconds
PS C:\Users\tejah> Set-ExecutionPolicy Bypass -Scope Process -Force
PS C:\Users\tejah> [System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072
PS C:\Users\tejah> iex ((New-Object
System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
WARNING: 'choco' was found at 'C:\ProgramData\chocolatey\bin\choco.exe'.
WARNING: An existing Chocolatey installation was detected. Installation will not continue.
This script will not
overwrite existing installations.
```

If there is no Chocolatey installation at 'C:\ProgramData\chocolatey', delete the folder and attempt the installation

again.

Please use choco upgrade chocolatey to handle upgrades of Chocolatey itself.

If the existing installation is not functional or a prior installation did not complete, follow these steps:

- Backup the files at the path listed above so you can restore your previous installation if needed.

- Remove the existing installation manually.

- Rerun this installation script.

- Reinstall any packages previously installed, if needed (refer to the lib folder in the backup).

Once installation is completed, the backup folder is no longer needed and can be deleted.

PS C:\Users\tejah> @"

>> class install_git {

>> package { 'git':

>> ensure => installed,

>> provider => 'chocolatey',

>> }

>> }

>>

>> include install_git

>> "@ | Out-File -FilePath "C:\puppet\site.pp" -Encoding ASCII

PS C:\Users\tejah> puppet apply C:\puppet\site.pp

Notice: Compiled catalog for laptop-qup1o2oi in environment production in 1.40 seconds

Error: Parameter provider failed on Package[git]: Invalid package provider 'chocolatey' (file: C:/puppet/site.pp, line: 2)

PS C:\Users\tejah> Set-ExecutionPolicy Bypass -Scope Process -Force

PS C:\Users\tejah> [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072

PS C:\Users\tejah> iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))

WARNING: 'choco' was found at 'C:\ProgramData\chocolatey\bin\choco.exe'.

WARNING: An existing Chocolatey installation was detected. Installation will not continue. This script will not

overwrite existing installations.

If there is no Chocolatey installation at 'C:\ProgramData\chocolatey', delete the folder and attempt the installation

again.

Please use choco upgrade chocolatey to handle upgrades of Chocolatey itself.

If the existing installation is not functional or a prior installation did not complete, follow these steps:

- Backup the files at the path listed above so you can restore your previous installation if needed.

- Remove the existing installation manually.

- Rerun this installation script.

- Reinstall any packages previously installed, if needed (refer to the lib folder in the backup).

Once installation is completed, the backup folder is no longer needed and can be deleted.

PS C:\Users\tejah> choco --version

2.5.1

PS C:\Users\tejah> puppet module install chocolatey-chocolatey

Notice: Preparing to install into C:/Users/tejah/.puppetlabs/etc/code/modules ...

Notice: Created target directory C:/Users/tejah/.puppetlabs/etc/code/modules

Notice: Downloading from https://forgeapi.puppet.com ...

Notice: Installing -- do not interrupt ...

C:/Users/tejah/.puppetlabs/etc/code/modules

└─┬ chocolatey-chocolatey (v1.2.6)

├── badgerious-windows_env (v2.2.2)

├── puppetlabs-powershell (v2.3.0)

└── puppetlabs-stdlib (v4.25.1)

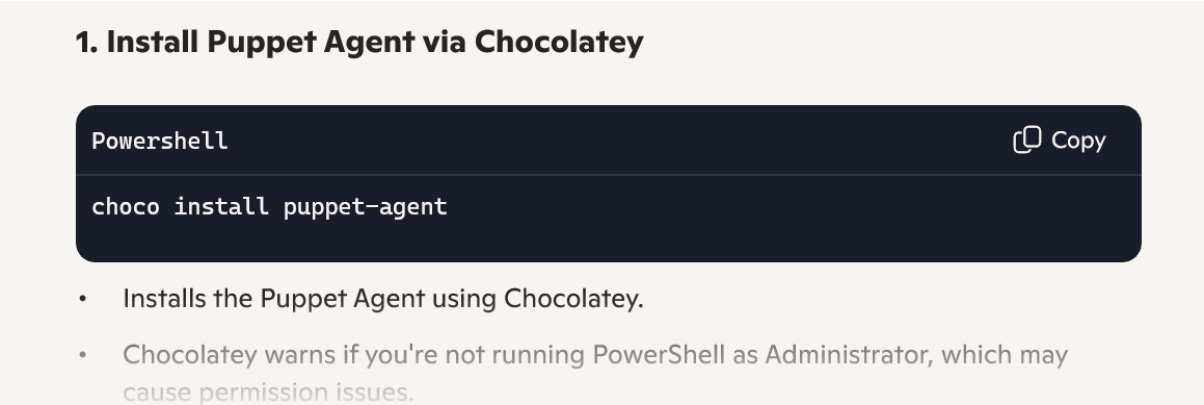
PS C:\Users\tejah> @"

>> class chocolatey_setup {

>> exec { 'install_chocolatey':

>> command => 'Set-ExecutionPolicy Bypass -Scope Process -Force; iex ((New-Object System.Net.WebClient).DownloadString("https://community.chocolatey.org/install.ps1"))',

```
>>     provider  => powershell,
>>     unless    => 'choco --version',
>>   }
>> }
>>
>> class install_git {
>>   package { 'git':
>>     ensure   => installed,
>>     provider => 'chocolatey',
>>   }
>> }
>>
>> include chocolatey_setup
>> include install_git
>> "@ | Out-File -FilePath "C:\puppet\site.pp" -Encoding ASCII
```



2. Create Puppet Manifest Directory

```powershell
New-Item -Path "C:\puppet" -ItemType Directory -Force
```

- Creates a folder `C:\puppet` to store your Puppet manifest (`site.pp`).
- `-Force` ensures the command succeeds even if the folder already exists.

3. Create Initial Manifest File

```powershell
Set-Content -Path "C:\puppet\site.pp" -Value @"
class install_git {
  package { 'git':
    ensure => installed,
  }
}

include install_git
"@
```

4. Apply Manifest

```powershell
puppet apply C:\puppet\site.pp
```

- Executes the manifest.

5. Prepare for Chocolatey Installation

```powershell
Set-ExecutionPolicy Bypass -Scope Process -Force
[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePoint
iex ((New-Object System.Net.WebClient).DownloadString('https://community.choc
```

6. Update Manifest to Use Chocolatey

```powershell
@"
class install_git {
  package { 'git':
    ensure   => installed,
    provider => 'chocolatey',
  }
}

include install_git
"@ | Out-File -FilePath "C:\puppet\site.pp" -Encoding ASCII
```

- Updates `site.pp` to specify `chocolatey` as the package provider.

7. Install Chocolatey Puppet Module

```powershell
puppet module install chocolatey-chocolatey
```

- Installs the official Puppet module for Chocolatey from Puppet Forge.