PRACTICAL NO. 6

Configuration Management

LOB4 To learn what containers are, using Docker, and the benefits they offer in terms of consistency, scalability, and efficiency.

LO4 Implement and evaluate containerized applications using Docker.

Software Configuration Management (SCM) is the process of systematically managing changes to software to maintain integrity and traceability throughout the software development lifecycle.

What is Ansible?

- Ansible is an open-source IT automation tool used for configuration management, application deployment, and orchestration.
- It automates repetitive tasks like setting up servers, installing packages, managing configurations, and deploying applications.
- Ansible is agentless, meaning it does not require agents installed on managed nodes.
- It uses SSH for Linux and WinRM for Windows machines.
- Ansible was initially designed to manage Linux/Unix environments using SSH for communication. This is why its core focus and tooling are naturally Linux-centric.

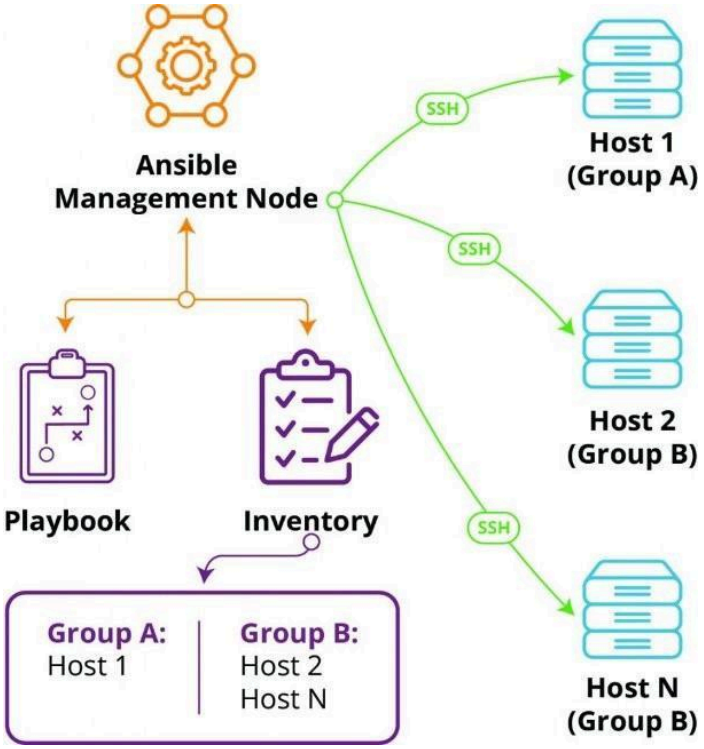
Where is Ansible Used?

- **Infrastructure as Code (IaC)**: Automating infrastructure provisioning
- **Application Deployment**: Managing multi-tier applications
- **Configuration Management**: Ensuring systems have the correct settings
- **Orchestration**: Managing complex workflows (e.g., scaling applications)
- **Security Compliance**: Enforcing security policies

How does Ansible work?

- Ansible uses the concepts of -
    - control nodes and
    - managed nodes.
- It connects from the control node, any machine with Ansible installed, to the managed nodes sending commands and instructions to them.
- The units of code that Ansible executes on the managed nodes are called modules.
- Each module is invoked by a task, and an ordered list of tasks together forms a playbook.
- Users write playbooks with tasks and modules to define the desired state of the system.
- The managed machines are represented in a simplistic inventory file that groups all the nodes into different categories.
- Ansible leverages a very simple language, YAML, to define playbooks in a human-readable data format that is easy to understand.
- Ansible doesn’t require the installation of any extra agents on the managed nodes so it’s simple to start using it.

- Typically, the only thing a user needs is a terminal to execute Ansible commands and a text editor to define the configuration files.



Ansible Ad-hoc commands -

- In Ansible, ad-hoc commands are
  - single, one-line commands
  - used to quickly execute tasks on remote hosts
  - without writing a playbook,
  - providing a quick and easy way to manage infrastructure.
- An Ansible ad hoc command uses the */usr/bin/ansible* command-line tool to automate a single task on one or more managed nodes.
- The ad hoc commands are quick and easy, but they are not reusable.
- Syntax -

ansible <target> -m <module> -a "<arguments>" [-b]

Where -

ansible - The command to invoke Ansible.

<target> - The inventory group or host on which to run the command (e.g., local, all, webserver).

-m <module> - The module to execute (e.g., file, user, apt).

-a "<arguments>" - The arguments for the module (in quotes).

-b - (Optional) Runs the command with sudo (root/super user) privileges.

Different (-m) Module Types in Ansible

Category	Common Modules
Command Execution	`command`, `shell`, `raw`
File Management	`file`, `copy`, `fetch`, `template`
User Management	`user`, `group`, `authorized_key`
Package Management	`apt`, `yum`, `dnf`, `pip`
Service Management	`service`, `systemd`, `cron`
Networking	`firewalld`, `ufw`, `iptables`
Cloud & Virtualization	`docker_container`, `aws_s3`, `gcp_compute_instance`
System Info	`ping`, `setup`, `hostname`
Databases	`mysql_db`, `postgresql_db`
Security	`selinux`, `pam_limits`

What is Ansible Playbook?

- An Ansible Playbook is a YAML file that defines a set of automation tasks for managing and configuring systems.
- Playbooks contain plays, which specify tasks that are executed on target systems (hosts).
- An Ansible Playbook is a YAML-based script that automates complex IT tasks, including configuration management, application deployment, and orchestration of multiple tasks across multiple systems.
- It defines what actions need to be performed on which machines in a structured and reusable way.
- Key Components of a Playbook:
  - Hosts: Defines target machines (inventory).
  - Tasks: Actions to be performed (e.g., installing software, modifying files).
  - Modules: Built-in commands like yum, apt, copy, file, etc.
  - Variables: Store reusable values for dynamic configurations.
  - Handlers: Execute tasks only when a change occurs (e.g., restart a service).
- Use the "vi firstplaybook.yml" command to create a playbook in user directory and add the following contents -

```
- name: First Ansible Playbook
  hosts: localhost
  gather_facts: no
  tasks:
        name: Print Hello World
        debug:
              msg: "Hello, World!"
```

- To run and view the output use – **ansible-playbook firstplaybook.yml**

Exercise:

1. Run Ad-hoc Commands for the following tasks -

```
student@admin: /mnt/c/Windows/System32
Microsoft Windows [Version 10.0.26100.3775]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>wsl -v -l
WSL version: 2.4.13.0
Kernel version: 5.15.167.4-1
WSLg version: 1.0.65
MSRDC version: 1.2.5716
Direct3D version: 1.611.1-81528511
DXCore version: 10.0.26100.1-240331-1435.ge-release
Windows version: 10.0.26100.3775

C:\Windows\System32>wsl -l -v
  NAME                   STATE           VERSION
* docker-desktop         Running         2
  Ubuntu-22.04           Stopped         2
```

   a. Execute an ad-hoc command to check the disk usage on remote hosts.

```
student@admin:/mnt/c/Windows/System32$ ansible all -m shell -a "df -h"
localhost | CHANGED | rc=0 >>
Filesystem      Size  Used Avail Use% Mounted on
/dev/sde       1007G  2.2G  954G   1% /
rootfs          1.8G  2.4M  1.8G   1% /init
none            1.8G  124K  1.8G   1% /dev/shm
tmpfs           4.0M     0  4.0M   0% /sys/fs/cgroup
none            1.8G  516K  1.8G   1% /run
none            1.8G     0  1.8G   0% /run/lock
none            1.8G   76K  1.8G   1% /run/user
drivers         225G  208G   17G  93% /usr/lib/wsl/drivers
none            1.8G     0  1.8G   0% /usr/lib/wsl/lib
none            1.8G  4.0K  1.8G   1% /mnt/wsl
none            1.8G  632K  1.8G   1% /mnt/wsl/docker-desktop/shared-sockets/host-services
/dev/sdc       1007G   57M  956G   1% /mnt/wsl/docker-desktop/docker-desktop-user-distro
/dev/loop0      482M  482M     0 100% /mnt/wsl/docker-desktop/cli-tools
none            1.8G   96K  1.8G   1% /mnt/wslg/versions.txt
none            1.8G   96K  1.8G   1% /mnt/wslg/doc
none            1.8G     0  1.8G   0% /usr/lib/modules/5.15.167.4-microsoft-standard-WSL2
C:\             225G  208G   17G  93% /mnt/c
D:\             250G   83G  168G  34% /mnt/d
```

   b. Fetch system uptime using the command module.

```
student@admin:/mnt/c/Windows/System32$ ansible all -m command -a "uptime"
localhost | CHANGED | rc=0 >>
 21:05:33 up  2:01,  1 user,  load average: 0.02, 0.01, 0.00
```

c. Retrieve the hostname of managed nodes using the setup module.

```
student@admin:/mnt/c/Windows/System32$ ansible all -m setup -a 'filter=ansible_hostname'
localhost | SUCCESS => {
    "ansible_facts": {
        "ansible_hostname": "admin",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

d. Create a directory */opt/myproject* on remote hosts.

```
student@admin:/mnt/c/Windows/System32$ ansible all -m file -a "path=/opt/myproject state=directory mode=0755" --become --ask-become-pass
BECOME password:
localhost | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "gid": 0,
    "group": "root",
    "mode": "0755",
    "owner": "root",
    "path": "/opt/myproject",
    "size": 4096,
    "state": "directory",
    "uid": 0
}
student@admin:/mnt/c/Windows/System32$
```

e. Ensure a file */opt/myproject/readme.txt* exists with specific permissions.

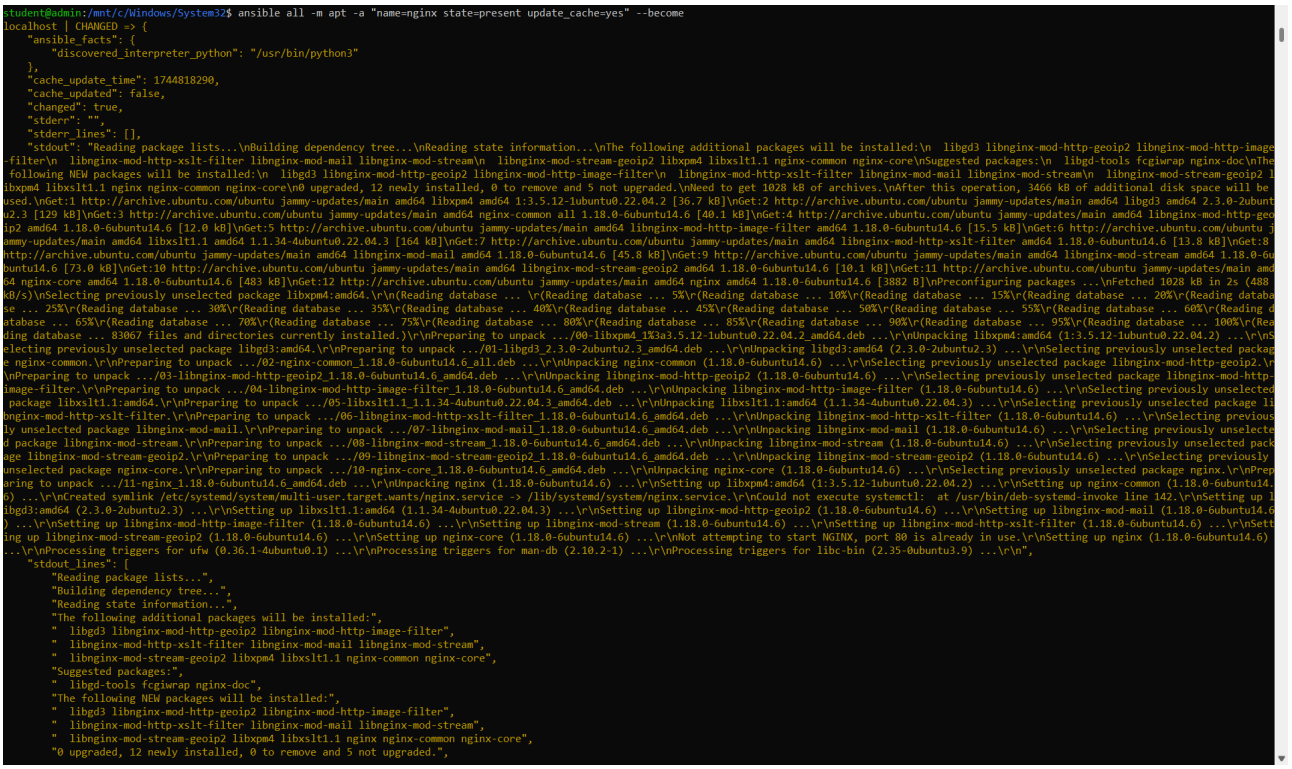
```
student@admin:/mnt/c/Windows/System32$ ansible all -m file -a "path=/opt/myproject/readme.txt state=touch mode=0644" --become --ask-become-pass
BECOME password:
localhost | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "dest": "/opt/myproject/readme.txt",
    "gid": 0,
    "group": "root",
    "mode": "0644",
    "owner": "root",
    "size": 0,
    "state": "file",
    "uid": 0
}
```

f. Create a new user *devops* with a home directory.

```
student@admin:/mnt/c/Windows/System32$ ansible all -m user -a "name=devops state=present create_home=yes" --become --ask-become-pass
BECOME password:
localhost | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "comment": "",
    "create_home": true,
    "group": 1001,
    "home": "/home/devops",
    "name": "devops",
    "shell": "/bin/sh",
    "state": "present",
    "system": false,
    "uid": 1001
}
```

g. Ensure *apache2 or httpd* is installed and running.

h. Install *nginx* on Ubuntu using the *apt* module.



i. Run a shell script located in */home/devops/setup.sh*

2. Create a Simple Playbook to Print "Hello World" using the pull model.

```
student@admin:/home$ ls -l
total 16
drwxr-x--- 2 devops  devops  4096 Apr 16 21:20 devops
drwxr-xr-x 3 root    root    4096 Apr  5 13:33 myansibleplaybook
drwxr-xr-x 2 root    root    4096 Apr 11 17:16 myansibleproj
drwxr-x--- 6 student student 4096 Apr 16 21:54 student
```

```
student@admin:/home$ sudo mkdir -p /home/Practical6Q2
[sudo] password for student:
student@admin:/home$ ls
Practical6Q2  devops  myansibleplaybook  myansibleproj  student
```

```
student@admin:/home$ cd Practical6Q2
student@admin:/home/Practical6Q2$
```

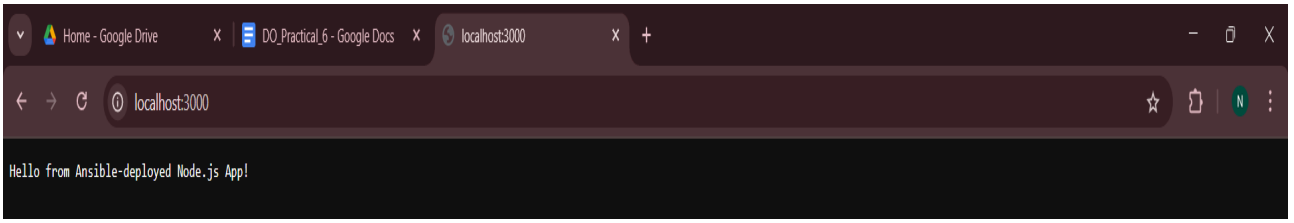
```
student@admin:/home/Practical6Q2$ sudo vi Practical6Q2.yml
student@admin:/home/Practical6Q2$ cat Practical6Q2.yml

---
- name: Practical No06 Q2 Ansible Practical
  hosts: localhost
  gather_facts: no
  tasks:
   - name: print hello world
     debug:
        msg: "Hello World From The Practical6Q2 From Ansible is SAY HELLO WOLRD!!!"
```

```
student@admin:/home/Practical6Q2$ ansible-playbook Practical6Q2.yml

PLAY [Practical No06 Q2 Ansible Practical] ***************************************

TASK [print hello world] *********************************************************
ok: [localhost] => {
    "msg": "Hello World From The Practical6Q2 From Ansible is SAY HELLO WOLRD!!!"
}

PLAY RECAP ***********************************************************************
localhost                  : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

3. Deploy a NodeJS web application with a basic welcome message using the pull model.

```
student@admin:/home/Practical6Q3$ cat Practical6Q3.yml

---
- name: Deploy Simple Node.js App without Git
  hosts: all
  become: yes

  tasks:
    - name: Install Node.js and npm
      apt:
        name:
          - nodejs
          - npm
        state: present
        update_cache: yes

    - name: Create app directory
      file:
        path: /opt/nodeapp
        state: directory
        mode: '0755'

    - name: Create app.js with Welcome Message
      copy:
        dest: /opt/nodeapp/app.js
        content: |
          const http = require('http');
          const port = 3000;
          const server = http.createServer((req, res) => {
            res.statusCode = 200;
            res.setHeader('Content-Type', 'text/plain');
            res.end('Hello from Ansible-deployed Node.js App!');
          });
          server.listen(port, () => {
            console.log(`Server running at http://localhost:${port}/`);
          });

    - name: Start Node.js App in background
      shell: nohup node app.js &
      args:
        chdir: /opt/nodeapp
```

```
student@admin:/home/Practical6Q3$ sudo ansible-playbook Practical6Q3.yml
[sudo] password for student:

PLAY [Deploy Simple Node.js App without Git] ***********************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [localhost]

TASK [Install Node.js and npm] *************************************************************************************************************************************************************************
changed: [localhost]

TASK [Create app directory] ****************************************************************************************************************************************************************************
changed: [localhost]

TASK [Create app.js with Welcome Message] **************************************************************************************************************************************************************
changed: [localhost]

TASK [Start Node.js App in background] *****************************************************************************************************************************************************************
changed: [localhost]

PLAY RECAP *********************************************************************************************************************************************************************************************
localhost                  : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



4. Create a simple playbook that demonstrates configuration management.

```
student@admin:/home/Practical6Q3$ cd
student@admin:~$ cd /home
student@admin:/home$ ls
Practical6Q2  Practical6Q3  Practical6Q4  devops  myansibleplaybook  myansibleproj  student
student@admin:/home$ cd Practical6Q4
student@admin:/home/Practical6Q4$ ls
Practical6Q4.yml  custom_apache.conf
```

```
student@admin:/home/Practical6Q4$ sudo vi Practical6Q4.yml
student@admin:/home/Practical6Q4$ cat Practical6Q4.yml
---
- name: Configuration Management Example
  hosts: localhost
  connection: local
  become: yes

  tasks:
    - name: Ensure user "tejas" exists
      user:
        name: tejas
        state: present
        shell: /bin/bash
        create_home: yes

    - name: Ensure directory /opt/project exists
      file:
        path: /opt/project
        state: directory
        mode: '0755'

    - name: Ensure /opt/project/config.txt file exists with content
      copy:
        dest: /opt/project/config.txt
        content: |
          # Project Configurations
          app_mode = "development"
          version = 1.0
        mode: '0644'

    - name: Install and start nginx
      apt:
        name: nginx
        state: present
        update_cache: yes

    - name: Ensure nginx is running and enabled
      service:
        name: nginx
        state: started
```

```
student@admin:/home/Practical6Q4$  ansible-playbook Practical6Q4.yml

PLAY [Configuration Management Example] ****************************************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************************************************
ok: [localhost]

TASK [Ensure user "tejas" exists] **********************************************************************************************************************************************************************
changed: [localhost]

TASK [Ensure directory /opt/project exists] ************************************************************************************************************************************************************
changed: [localhost]

TASK [Ensure /opt/project/config.txt file exists with content] *****************************************************************************************************************************************
changed: [localhost]

TASK [Install and start nginx] *************************************************************************************************************************************************************************
ok: [localhost]

TASK [Ensure nginx is running and enabled] *************************************************************************************************************************************************************
ok: [localhost]

PLAY RECAP *********************************************************************************************************************************************************************************************
localhost                  : ok=6    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

5.Integrate Ansible with Jenkins for CI/CD wherein write a playbook to deploy an application and configure Jenkins to execute the playbook after each code push.

```
venom@VENOM:/home/practical06q5$ cat deploy.yml
---
- name: Local Node App Deployment
  hosts: local
  connection: local
  become: yes

  tasks:
    - name: Ensure app folder exists
      file:
        path: /opt/myapp
        state: directory

    - name: Create index.js file
      copy:
        dest: /opt/myapp/index.js
        content: |
```

- name: Local Node App Deployment

  hosts: local

  connection: local

  become: yes

  tasks:

    - name: Ensure app folder exists

      file:

        path: /opt/myapp

        state: directory

    - name: Create index.js file

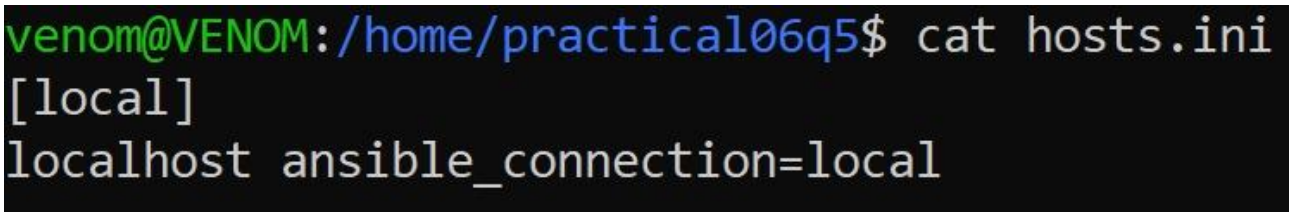
      copy:

        dest: /opt/myapp/index.js

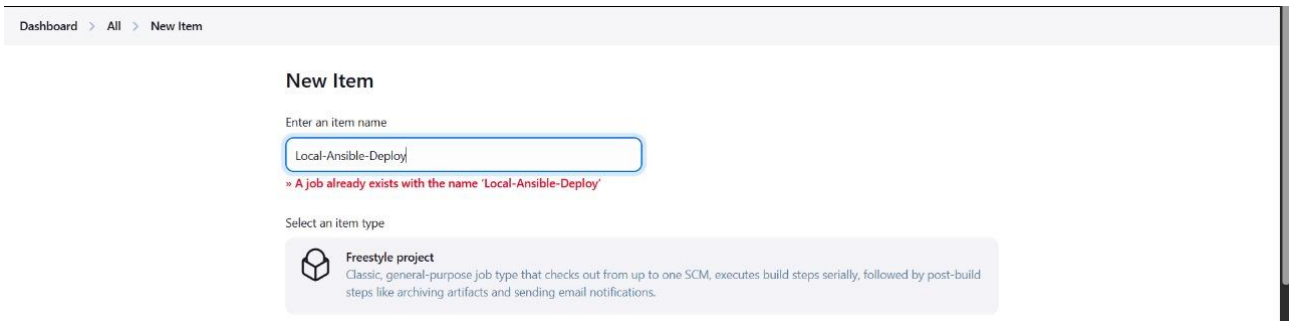
        content: |

          const http = require('http');

```
    const server = http.createServer((req, res) => {
      res.end(" 🚀 Hello from Jenkins + Ansible!");
    });
    server.listen(3000);
- name: Install Node.js
  apt:
    name: nodejs
    state: present
    update_cache: yes
- name: Start app in background
  shell: "nohup node index.js > app.log 2>&1 &"
  args:
    chdir: /opt/myapp
```




```
[local]
localhost ansible_connection=local
```



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

```
wsl -d Ubuntu-22.04 ansible-playbook -i /home/practical06q5/hosts.ini /home/practical06q5/deploy.yml
```

Advanced

Add build step

Console Output

```
Started by user TEJAS MAHESH BHIDE
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\Local-Ansible-Deploy
[Local-Ansible-Deploy] $ cmd /c call C:\Users\ASUS\AppData\Local\Temp\jenkins7630090728855269772.bat

C:\ProgramData\Jenkins\.jenkins\workspace\Local-Ansible-Deploy>wsl -d Ubuntu-22.04 ansible-playbook -i /home/practical06q5/hosts.ini /home/practical06q5/deploy.yml

PLAY [Local Node App Deployment] ***********************************************

TASK [Gathering Facts] *********************************************************
ok: [localhost]

TASK [Ensure app folder exists] ************************************************
ok: [localhost]

TASK [Create index.js file] ****************************************************
ok: [localhost]

TASK [Install Node.js] *********************************************************
ok: [localhost]

TASK [Start app in background] *************************************************
changed: [localhost]

PLAY RECAP *********************************************************************
localhost                  : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0


C:\ProgramData\Jenkins\.jenkins\workspace\Local-Ansible-Deploy>exit 0
Finished: SUCCESS
```