Sample Ansible Tasks for DevOps

Author: Zayan Ahmed | **Estimated Reading time:** 5 mins

Ansible is a tool that helps us make computers do things automatically. In DevOps, this means using Ansible to set up servers, deploy apps, and fix problems. Let's look at some easy-to-understand scenarios and how Ansible can help solve them.

Scenario 1: Setting Up a Web Server

Imagine you have a new computer, and you want it to show websites. But first, you need to install a program called Apache to make it work as a web server.

Ansible Task

Here's an Ansible playbook to install Apache and make sure it starts automatically:

```yaml
- name: Install and start Apache
  hosts: web_servers
  become: yes
  tasks:
    - name: Install Apache
      apt:
        name: apache2
        state: present
```

```
  - name: Start Apache service
    service:
      name: apache2
      state: started
      enabled: yes
```

Scenario 2: Deploying Code to a Server

You wrote a cool app and need to copy it to your server so everyone can use it. But doing this by hand takes too long.

Ansible Task

This playbook copies your app files to the server and restarts the app:

```
- name: Deploy app to server
 hosts: app_servers
 become: yes
 tasks:
   - name: Copy app files
     copy:
       src: /path/to/your/app/
       dest: /var/www/myapp/

   - name: Restart app service
     service:
       name: myapp
       state: restarted
```

Scenario 3: Adding a New User

Your teammate needs access to the server, but you don’t want to manually create their account on every computer.

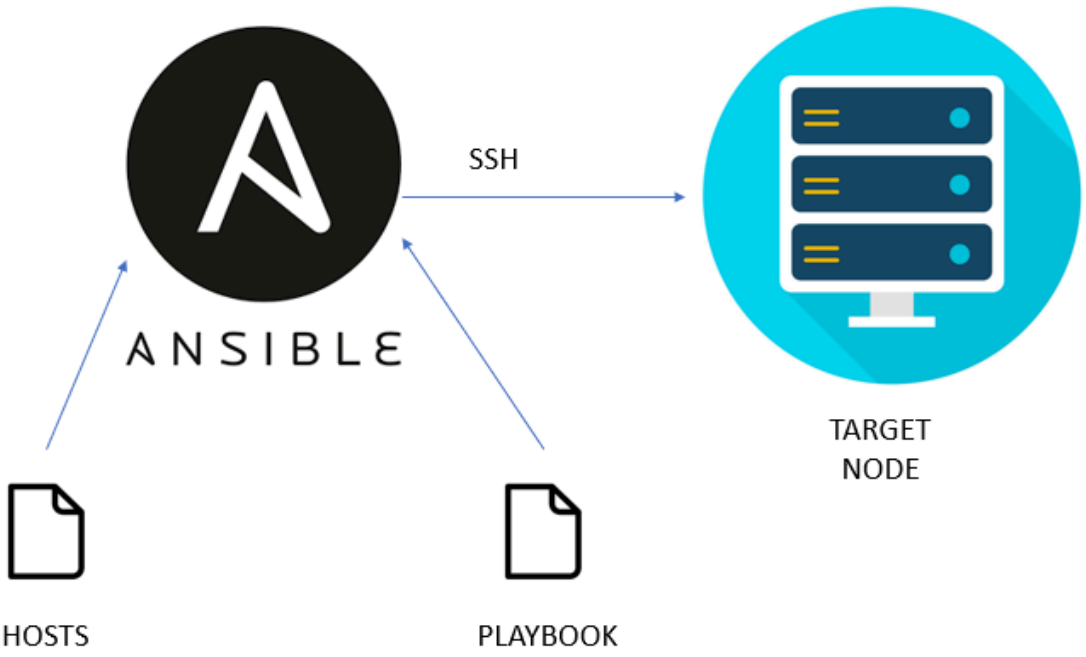
Ansible Task

This playbook creates a new user and sets their password:

```
- name: Add a new user
 hosts: all
 become: yes
 tasks:
   - name: Create user
```

```
    user:
      name: teammate
      state: present
      groups: sudo

  - name: Set user password
    shell: echo "teammate:password123" | chpasswd
```



Scenario 4: Checking Server Health

Sometimes, you need to check if your servers are running out of space or memory.

Ansible Task

This playbook runs a health check and saves the results:

```
- name: Check server health
 hosts: all
 become: yes
 tasks:
   - name: Check disk space
     command: df -h
     register: disk_space

   - name: Show disk space
```

```
    debug:
      var: disk_space.stdout

  - name: Check memory usage
    command: free -m
    register: memory_usage

  - name: Show memory usage
    debug:
      var: memory_usage.stdout
```

Scenario 5: Fixing a Broken App

An app stops working because its service isn't running. You need to restart it quickly.

Ansible Task

This playbook checks if the app service is running and starts it if it's not:

```
- name: Fix broken app service
 hosts: app_servers
 become: yes
 tasks:
   - name: Check app service
     service:
       name: myapp
       state: started
       enabled: yes
```

Conclusion

Ansible makes it super easy to handle repetitive tasks, like setting up servers, deploying code, and fixing problems. These simple playbooks are just a start. You can create more complex tasks as you learn and grow in DevOps!

Want more ?
Follow me on **LinkedIn**