COMPLETE SETUP GUIDE (Start to Finish)

Step 1: Install Ansible (Control Node)

```
# Ubuntu/Debian
sudo apt update && sudo apt install ansible sshpass -y

# RHEL/CentOS/Rocky
sudo yum install epel-release -y
sudo yum install ansible sshpass -y

# macOS
brew install ansible

# Windows (WSL2)
sudo apt update && sudo apt install ansible sshpass -y

# Verify Installation
ansible --version
```

Step 2: SSH Key Setup (One-Time)

```
# Generate SSH Key (if you don't have one)
ssh-keygen -t rsa -b 4096 -f ~/.ssh/ansible_key -N ""

# Copy to ONE test server first
ssh-copy-id -i ~/.ssh/ansible_key.pub user@your-server-ip

# Test Connection
ssh -i ~/.ssh/ansible_key user@your-server-ip
```

Step 3: Project Structure

```
# Create project folder
mkdir ansible-remote-scripts && cd ansible-remote-scripts

# Create all necessary files
touch ansible.cfg inventory.ini remote-script.yml deploy.sh
```

COMPLETE CODE FILES

1. ansible.cfg (Configuration)

ansible.cfg

```
[defaults]
inventory = ./inventory.ini
host_key_checking = False
gathering = smart
ansible_python_interpreter = /usr/bin/python3
private_key_file = ~/.ssh/ansible_key

[privilege_escalation]
become = True
become_method = sudo
become_user = root
become_ask_pass = False
```

2. `inventory.ini` (Your Server List)

inventory.ini

```
# Single server for testing
[test]
192.168.1.100

# Multiple servers
[webservers]
web1.example.com
web2.example.com
web3.example.com

[dbservers]
db1.example.com
db2.example.com

# Group variables
[webservers:vars]
ansible_user=ubuntu
ansible_ssh_private_key_file=~/.ssh/ansible_key

[all:vars]
ansible_python_interpreter=/usr/bin/python3
```

3. `deploy.sh` (Your Shell Script)

deploy.sh

```
#!/bin/bash
# Sample Deployment Script
echo "========================================"
echo " Deployment started on: $(hostname)"
echo " Timestamp: $(date)"
echo "========================================"

# System Information
echo "System Overview:"
echo "- CPU Cores: $(nproc)"
echo "- Memory: $(free -h | awk '/^Mem:/{print $2}')"
echo "- Disk: $(df -h / | awk 'NR==2{print $4}') free"

# Application Deployment Logic
echo "Installing dependencies..."
# Add your actual commands here
# sudo apt update
# sudo apt install -y nginx

echo "Deployment completed successfully!"
echo "Status: $(systemctl is-active nginx 2>/dev/null || echo 'Service check skipped')"
echo "========================================"
```

4. `remote-script.yml` (MAIN PLAYBOOK)

remote-script.yml

```yaml
---
- name: Execute Script on Remote Servers
  hosts: all  # Runs on ALL servers in inventory
  become: yes  # Run with sudo privileges

  tasks:
    # Upload script to remote server
    - name: Copy script to remote server
      ansible.builtin.copy:
        src: deploy.sh
        dest: /tmp/deploy.sh
        owner: root
        group: root
        mode: '0755'  # Make executable
      register: copy_result

    # Execute the script
    - name: Run deployment script
      ansible.builtin.shell:
        cmd: /tmp/deploy.sh
      register: script_output
      args:
        chdir: /tmp  # Change directory before running
      async: 45  # Timeout in seconds
      poll: 5    # Check every 5 seconds

    # Display beautiful output
    - name: Show execution results
      ansible.builtin.debug:
        msg: |
          ========================================
           Script executed on: {{ inventory_hostname }}
           Exit Code: {{ script_output.rc }}
           Duration: {{ script_output.delta }}
          ========================================
          {{ script_output.stdout }}
          ========================================

    #  Handle failures gracefully
    - name: Notify on failure
      ansible.builtin.debug:
        msg: " Script failed on {{ inventory_hostname }}"
      when: script_output.rc ≠ 0

    #  Cleanup (optional)
    - name: Remove script after execution
      ansible.builtin.file:
        path: /tmp/deploy.sh
        state: absent
      when: script_output.rc == 0
```

EXECUTION COMMANDS

Basic Execution

remote-script.yml

```
# Test connection first
ansible all -m ping

# Run on ALL servers
ansible-playbook remote-script.yml

# Run only on webservers group
ansible-playbook remote-script.yml --limit webservers

# Run on specific server
ansible-playbook remote-script.yml --limit 192.168.1.100
```

Dry Run (Test Only)

remote-script.yml

```
# See what would happen without making changes
ansible-playbook remote-script.yml --check --diff
```

Advanced Usage

remote-script.yml

```
# Run with verbose output (debugging)
ansible-playbook remote-script.yml -vvv

# Run with tags
ansible-playbook remote-script.yml --tags "deploy,cleanup"

# Parallel execution (10 servers at once)
ansible-playbook remote-script.yml -f 10

# Save output to file
ansible-playbook remote-script.yml | tee deployment.log
```

How You Can Use on Different Use Cases

Daily Automation Tasks

remote-script.yml

```yaml
# backup.yml - Backup databases on all servers
- name: Backup MySQL databases
  shell: mysqldump -u root --all-databases > /backups/db-$(date +%Y%m%d).sql
  become: yes

# update.yml - Security updates
- name: Update all servers
  apt:
    update_cache: yes
    upgrade: dist
  become: yes

# monitor.yml - Health checks
- name: Check disk space
  shell: df -h | grep -E '^(Filesystem|/[sv]da)'
  register: disk_space
```

Production Deployment Pipeline

remote-script.yml

```yaml
# 1. Deploy code
- command: git pull origin main
  args:
    chdir: /var/www/app

# 2. Install dependencies
- pip:
    requirements: /var/www/app/requirements.txt

# 3. Restart services
- systemd:
    name: gunicorn
    state: restarted
    daemon_reload: yes
```

TROUBLESHOOTING GUIDE

Error	Solution
"Permission denied"	`chmod 600 ~/.ssh/ansible_key`
Python not found	Install Python3 on remote: `sudo apt install python3-minimal`
SSH timeout	Check firewall: `sudo ufw allow 22`
Sudo password needed	Add user to sudoers without password
Host unreachable	Verify IP in inventory and network connectivity

BENEFITS & METRICS

Before Ansible	After Ansible	Improvement
10 servers × 5 min each	**1 command × 30 seconds**	**10x faster**
Manual errors common	**Consistent every time**	**100% accuracy**
No audit trail	**Full execution logs**	**Complete visibility**
One-by-one updates	**Parallel execution**	**Scalable to 1000+ servers**