

Ansible **AWX** is an open source web application that provides a user interface, REST API, and task engine for **Ansible**. It's the open source version of the **Ansible** Tower. The **AWX** allows you to manage **Ansible** playbooks, inventories, and schedule jobs to run using the web interface.

<https://github.com/ansible/awx>

### **System Requirements**

The system that runs the AWX service will need to satisfy the following requirements

* At least 4GB of memory
* At least 2 cpu cores
* At least 20GB of space
* Running Docker, Openshift, or Kubernetes
* If you choose to use an external PostgreSQL database, please note that the minimum version is 10+.
* OS
  + Ubuntu version - Ubuntu\_Server\_16.04\_LTS
  + CentOS -
  + RHEL -

### **Software Prerequisites**

Before you can run a deployment, you'll need the following installed in your local environment:

* [Ansible](http://docs.ansible.com/ansible/latest/intro_installation.html) Requires Version 2.8+
* [Docker](https://docs.docker.com/engine/installation/)
  + A recent version
* [docker](https://pypi.org/project/docker/) Python module
  + This is incompatible with docker-py. If you have previously installed docker-py, please uninstall it.
  + We use this module instead of docker-py because it is what the docker-compose Python module requires.
* [GNU Make](https://www.gnu.org/software/make/)
* [Git](https://git-scm.com/) Requires Version 1.8.4+
* [Node 10.x LTS version](https://nodejs.org/en/download/)
* [NPM 6.x LTS](https://docs.npmjs.com/)
* Python 3.6+

**Installation**:

Note: Awx installation steps on Ubuntu machine

**You can also use installation script :**

**========================================**

<https://github.com/RohitMalakarGslab/ansible_aws/blob/master/AWX_installation.sh>

**=========================================**

**Below are manual steps for installation**

$ sudo apt-get update

$ sudo apt update && sudo apt install python -y

Install Ansible

============================================================

$ sudo apt-get install software-properties-common

$ sudo apt-add-repository --yes --update ppa:ansible/ansible

$ sudo apt-get install ansible

$ ansible --version

Install docker

===========================================================

$ sudo apt-get install -y apt-transport-https ca-certificates curl gnupg-agent software-properties-common

$ curl -fsSL<https://download.docker.com/linux/ubuntu/gpg> | sudo apt-key add -

$ sudo add-apt-repository "deb [arch=amd64]<https://download.docker.com/linux/ubuntu> $(lsb\_release -cs) stable"

$ sudo apt-get update && sudo apt-get install docker-ce docker-ce-cli containerd.io

$ sudo systemctl status docker

Install Python-pip, Docker-compose

============================================================

$ sudo -i

$ apt install git python-pip -y

$ pip install docker-compose==1.9.0

Install NodeJS, NPM

============================================================

$ apt install nodejs npm -y

$ npm install npm --global

Install Git & Clone Repository AWX

============================================================

$ apt install git

$ cd /root

$ git clone https://github.com/ansible/awx.git

$ cd awx/installer/

Generate Secret Key & Edit Inventory File

============================================================

$ openssl rand -hex 32

qqdd4c70bbf82d572bfbdd454b28172642e6b8061b8c8b4937d9134294d30e09

Edit inventory file and change/add below parameters

============================================================

1# postgres\_data\_dir

postgres\_data\_dir=/var/lib/pgdocker

2# host\_port

host\_port=8080

3# use\_docker\_compose

use\_docker\_compose=true

4# docker\_compose\_dir

docker\_compose\_dir=/var/lib/awx

5# pg\_password

pg\_password=pgpass

6# rabbit\_mq

rabbitmq\_password=rabb\_pass

rabbitmq\_erlang\_cookie=rabb\_cookie

7# admin\_user\_password

admin\_user=admin

admin\_password=adminpass

8# secret\_key

secret\_key=<set the key you generated above>

9# project\_data\_dir

project\_data\_dir=/var/lib/awx/projects

10# change the python interpreter from python3 to python

Note: first line of inventory file

localhost ansible\_connection=local ansible\_python\_interpreter="/usr/bin/env python"

**Install AWX**

===============================================================

$ ansible-playbook -i inventory install.yml

**Check Services AWX:**

===============================================================

$ cd /var/lib/awx

$ docker-compose ps

==============================================================

**Install AWX cli**

**===============================================================**

**1. Login to the Linux instance which has network connectivity to reach ansible Tower/AWX.**

**2. Use the pip module to install tower-cli.**

**$ pip install ansible-tower-cli**

**3. Check tower-cli command to check the version.**

**$ tower-cli --version**

**4. Here is the help page of tower-cli**.

$ tower-cli --help

**5. Set the Ansible Tower or AWX URL to tower-cli config using the following command.**

$ tower-cli config host <http://localhost:8080>

**6. Set the privileged account for authenticating to Ansible Tower on Tower CLI**

**$ tower-cli config username admin**

**7. Set the admin password on Tower CLI for your ansible tower/AWXinstance.**

**$ tower-cli config password password**

**8. Set Verify SSL as false. (it might vary depending on your ansible Tower site protocal).**

**$ tower-cli config verify\_ssl False**

**9. Preview the config of tower-cli using the following command.**

**$ tower-cli config**

**10. How to confirm tower-cli is authenticating to Ansible tower successfully? Let’s fire a command to get all the recent job list. Tower CLI reaches Ansile tower’s API to fetch the required information.**

**$ tower-cli job list**

==============================================================

**Open awx UI**

==============================================================

http://localhost:8080

username: admin

password: password

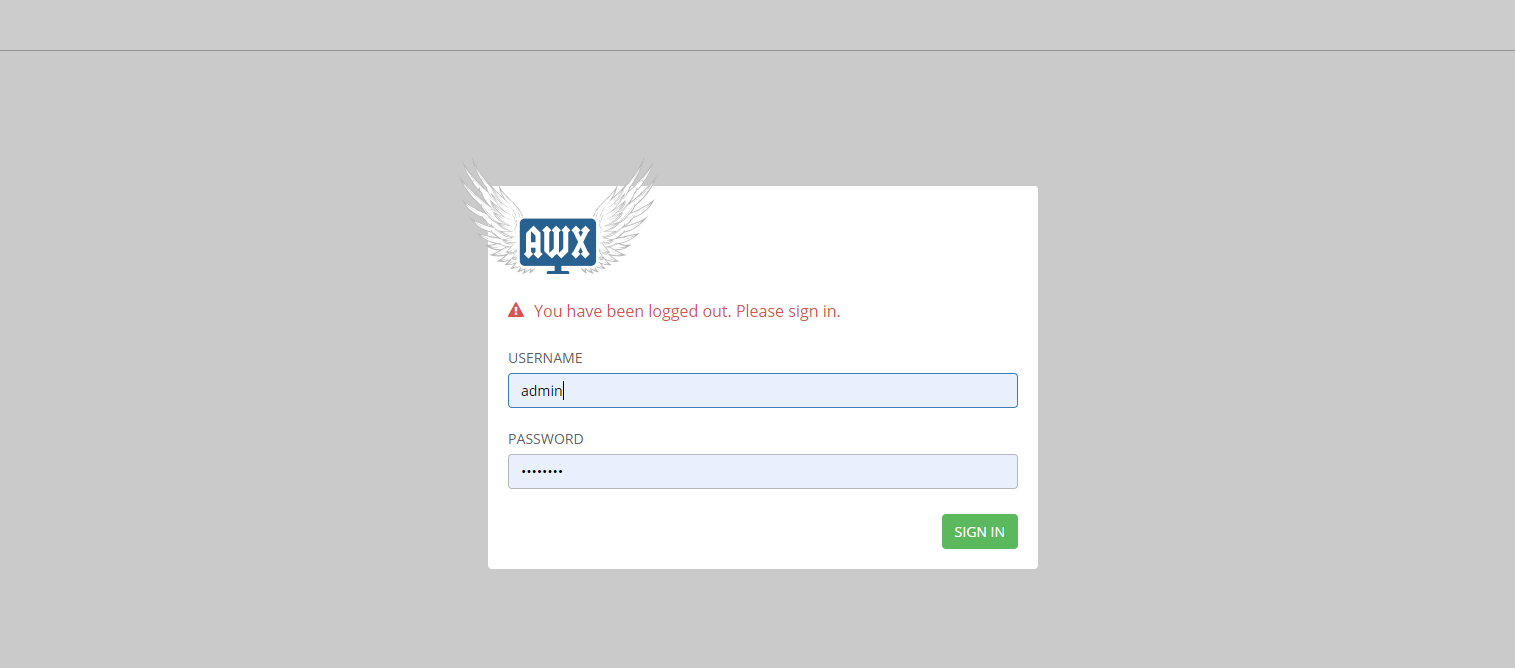
## Running a playbook on the AWX server

**Perform the following steps to run a playbook:**

1. **Log in to the AWX server with the default credentials (user name as admin and password as password).**
2. **Default username and password after installation of AWX are**

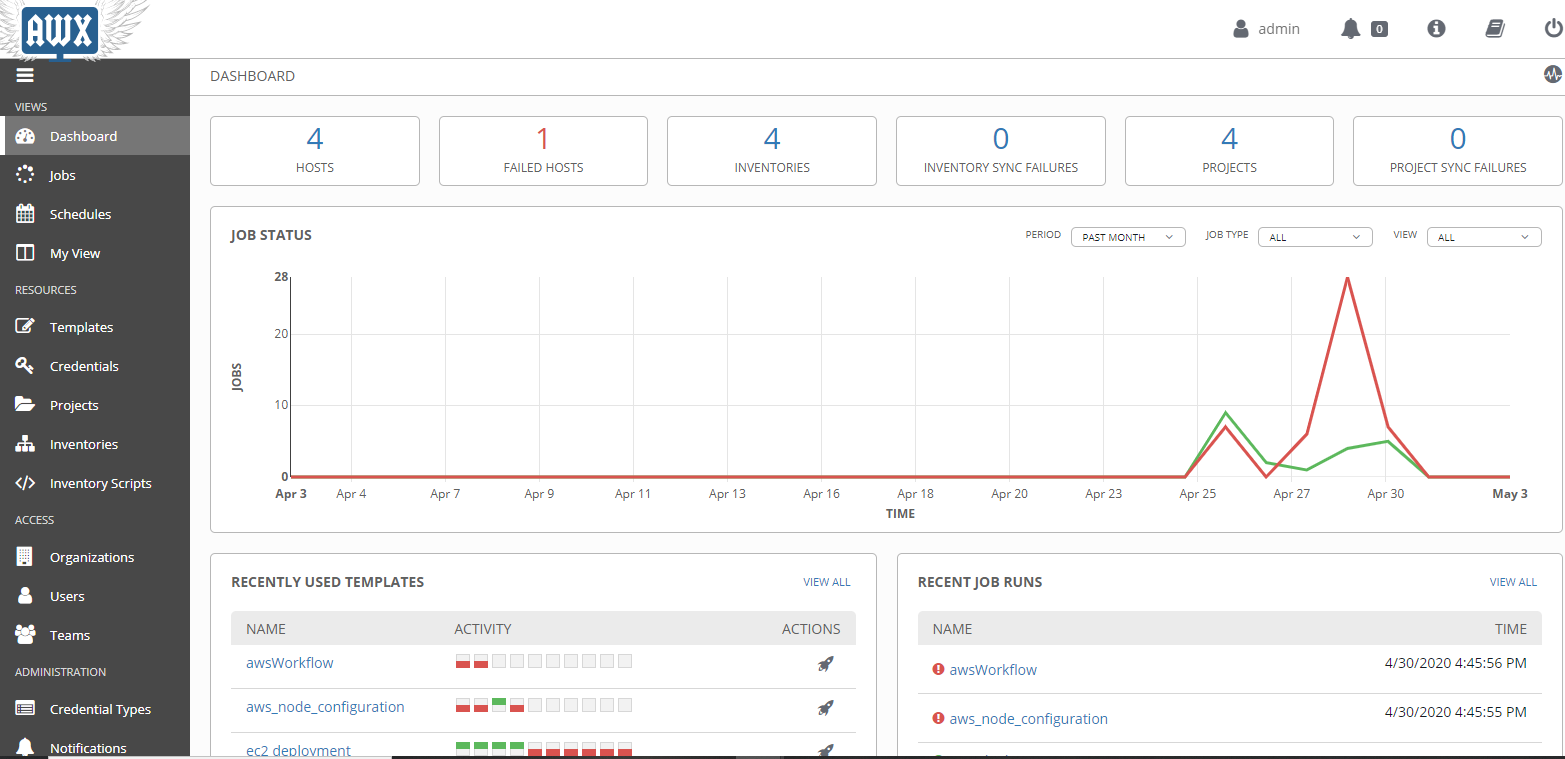
**Username**: admin

**password**: password

***Figure 1. Login Page***

On the home page or the dashboard, you can see information about your AWX server and its overall status that includes the following details:

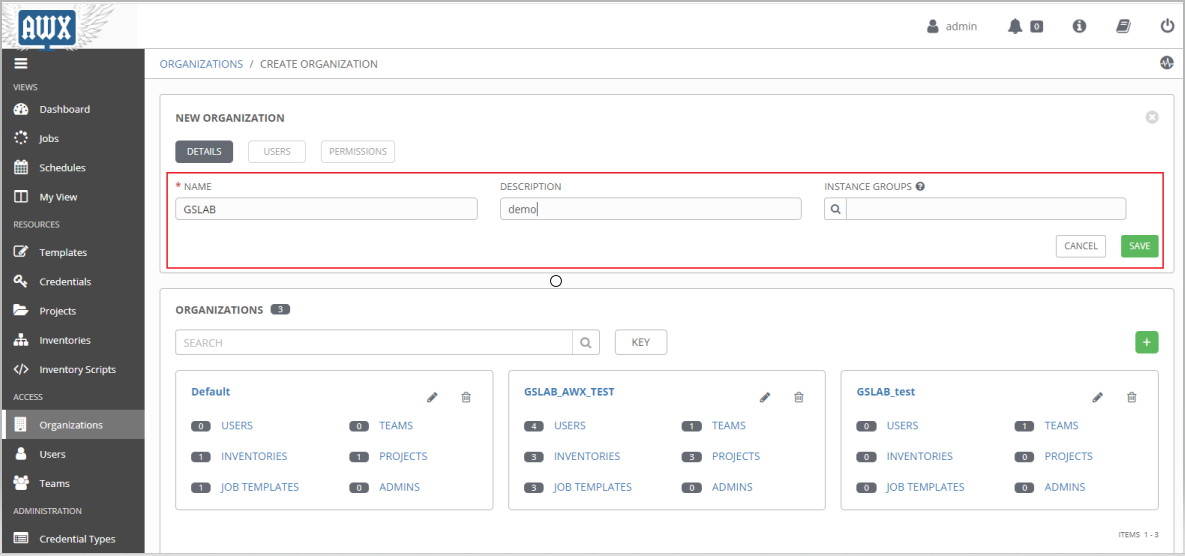
* Number of hosts who have successfully run the playbooks
* Number of hosts who failed to run the playbooks
* Total number of inventories
* Number of projects and the sync status
* Graph of playbook that has been run throughout

*Figure 2. Dashboard*

**2. Add a new organization.**

1. On the dashboard, click ORGANIZATIONS on the left pane.
2. Click ADD and enter a name and a brief description for the new organization.
3. Click SAVE to save changes.

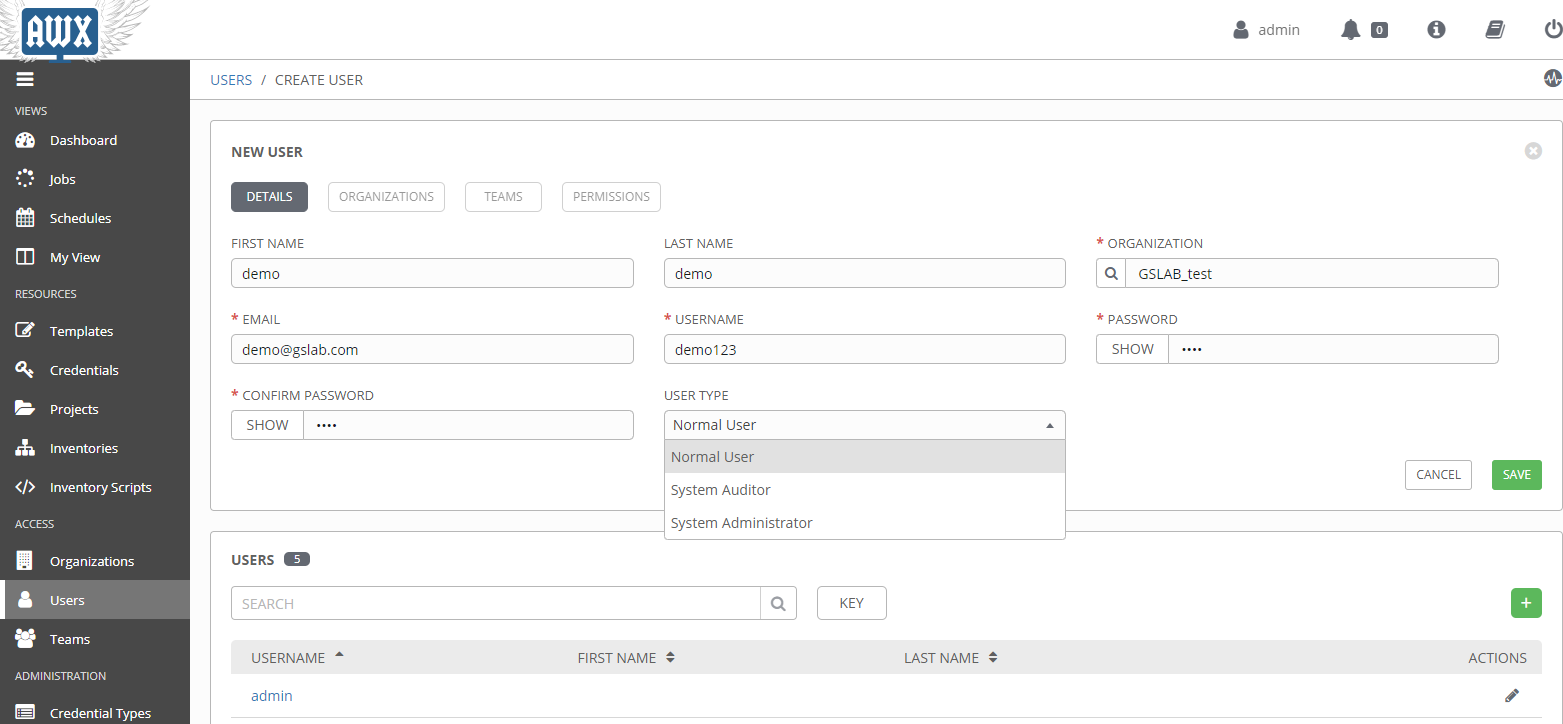
*Figure 3. Creating a new organization*

**

**3. Add a new user**

1. On the left pane, click USERS.
2. Enter the first name and the last name for the user.
3. Enter organization you created in the previous step.
4. Enter a working email ID and a specify the user name and password to log in to the server.
5. Select the User type as per the access rights to be given. Users can be of any one of the following types:
   * Normal user – is a member of an organization who can create new templates, use templates, and update templates.
   * System auditor – is a member of an organization who can view inventory, templates, and job status but cannot create or modify anything on the server.
   * System administrator – has all the privileges on the server (same as the default root/admin).
6. Click Save to save changes.

*Figure 4. Create a new user*

**

**Other authentication mechanism e.g**.

* Active Directory
* Google Authentication
* Etc..

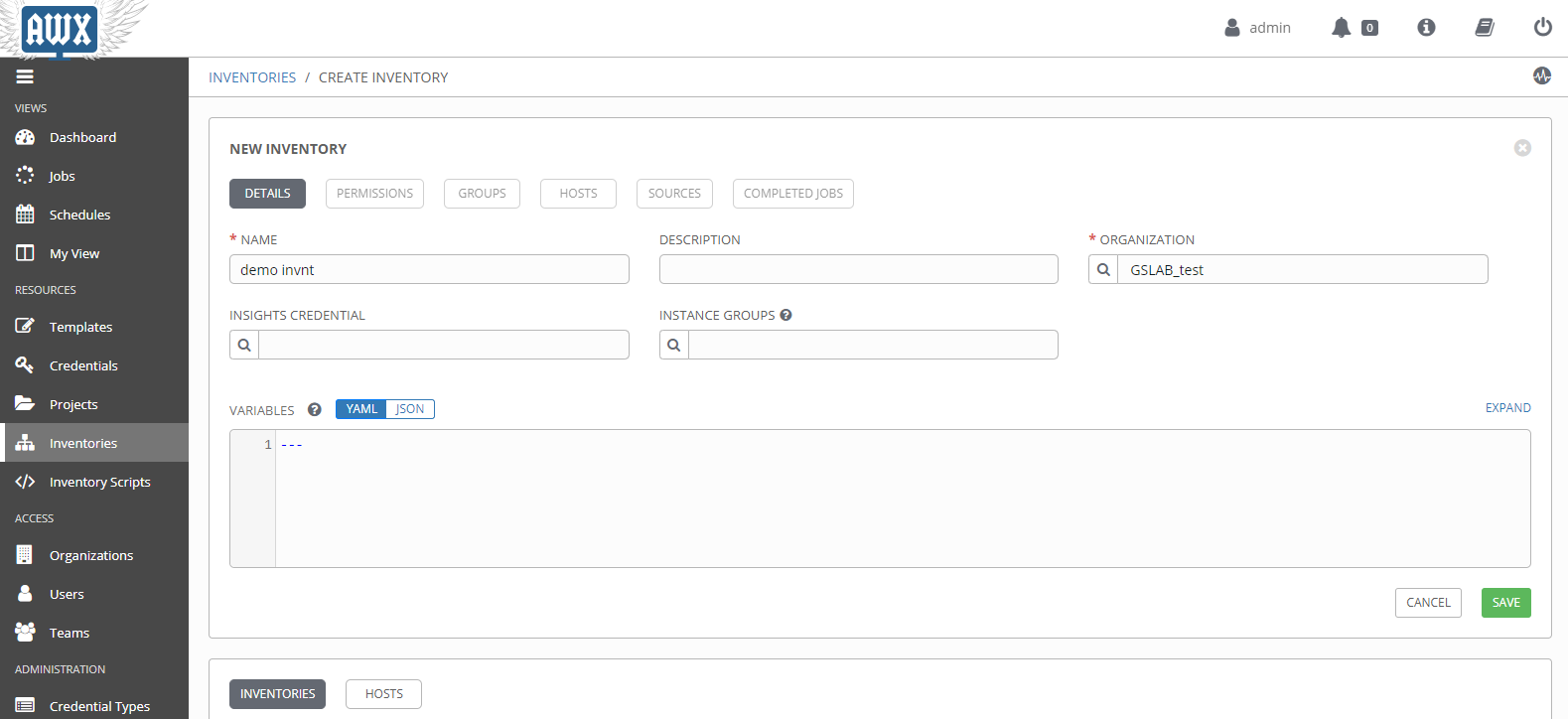
***4. Add a new inventory.***

*Adding an inventory is a task to add hosts to the server. All your remote hosts will come under inventory. An inventory can be divided into groups, such as development, testing, and production servers. To add a new inventory:*

1. *On the left pane, click INVENTORIES.*
2. *Click ADD INVENTORY.*
3. *Enter a name and specify an organization for the inventory.*
4. *Click SAVE to save changes.*

*Generic question - What hostname and IP address need to use to provision new VM (e.g. EC2 on AWS, as we don’t know the IP address unless it’s deployed)*

*Figure 5. Create a new inventory*

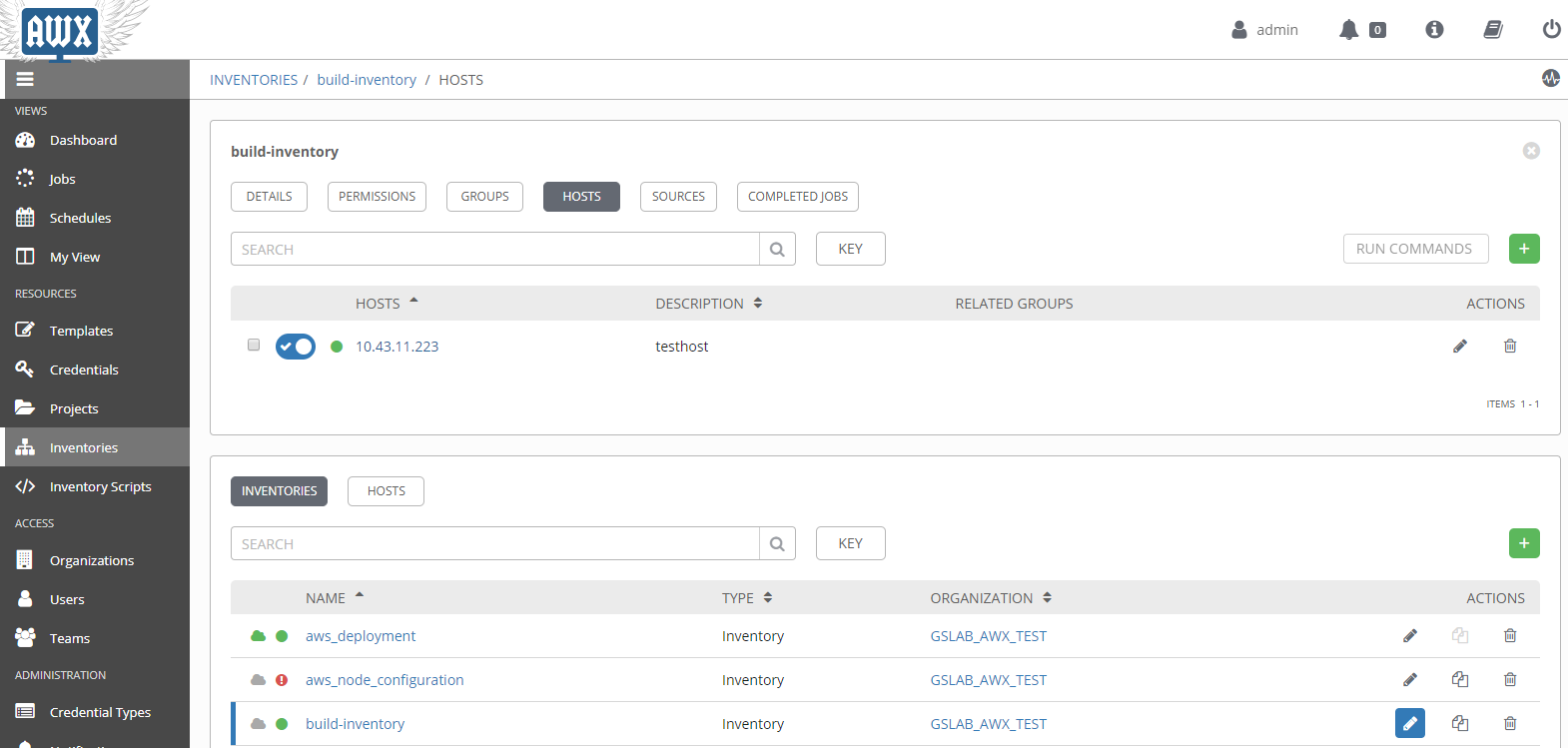
**

***5. Add hosts.***

*Host name can be a working IP address or a URL. For example, 192.168.1.23 or aaa.company.com.*

1. *In the same inventory page previously created, click the HOSTS tab.*
2. *Then click ADD HOST.*
3. *Enter the host name of the machine you need to add.*
4. *Enter a description of the machine.*
5. *Click SAVE to save changes. You can add any number of hosts to an inventory.*

*Figure 6. Add a host to inventory*

**

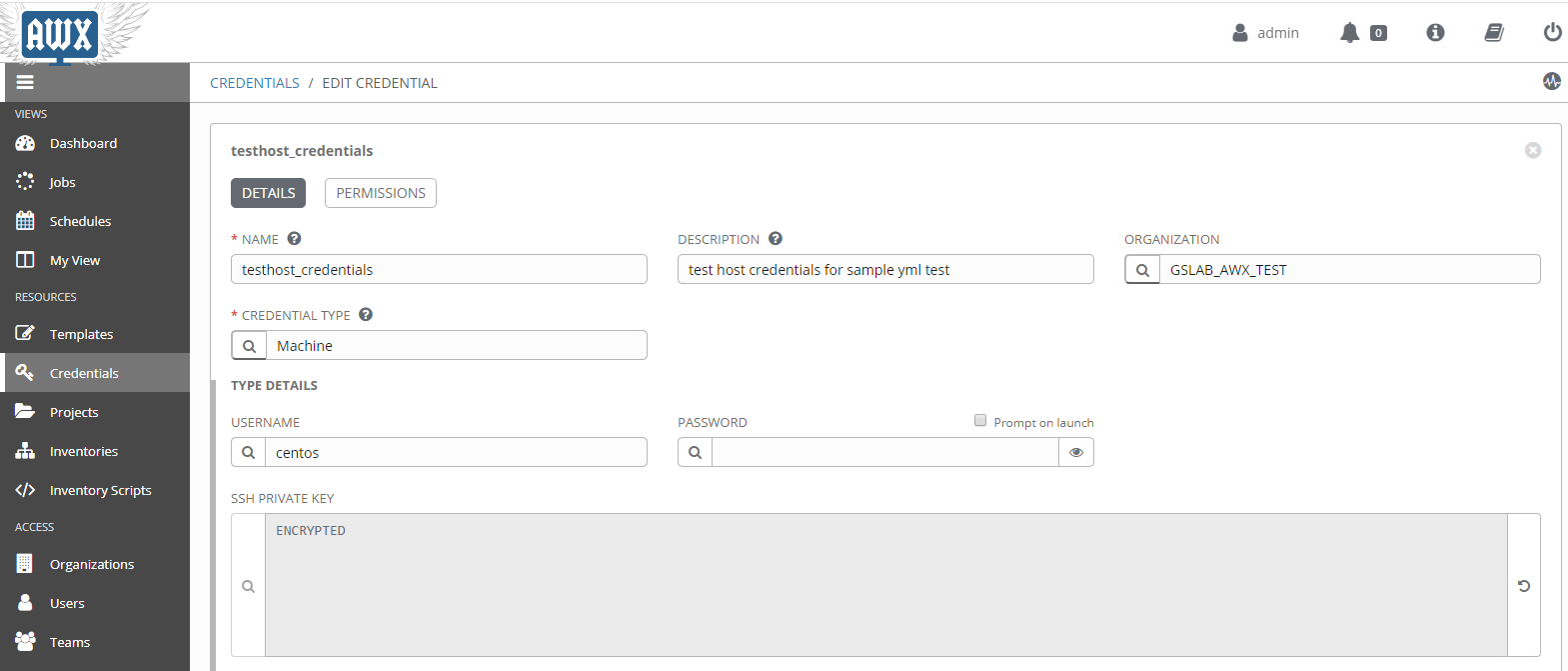
***6. Add credentials.***

*In AWX, credentials are stored separately. This is very efficient in a LDAP scenario where we can use a single credential to any number of hosts.*

*We will explore more on this part.*

1. *On the left pane, click CREDENTIALS.*
2. *Click ADD and enter a name and description for the new credential.*
3. *Select an organization for the credential.*
4. *Select a credential type (Machine – similar to SSH login).*
5. *Enter the username and password of the remote machine.*
6. *Click SAVE to save changes made.*

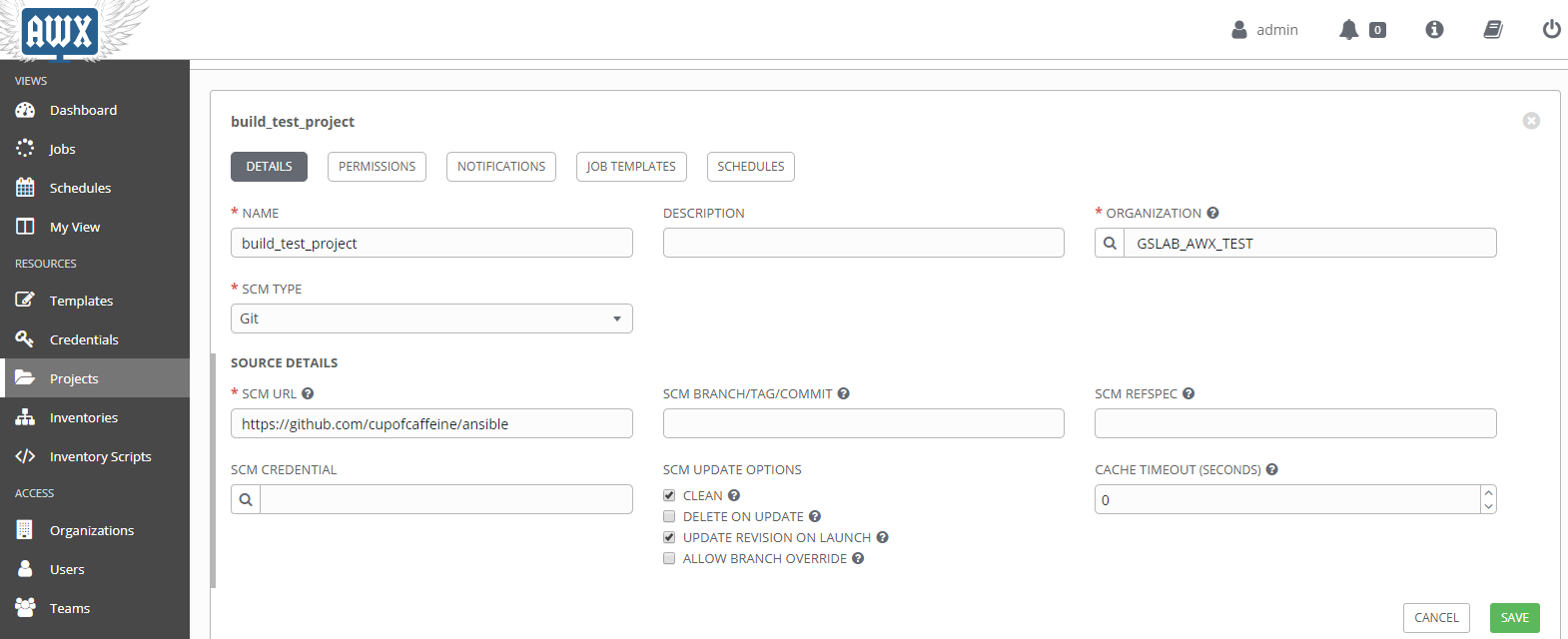
*Figure 7. Add credentials of corresponding host.*

**

***7. Add a new project.***

1. *On the left pane, click PROJECTS.*
2. *Click ADD.*
3. *Enter a name and description for the project.*
4. *Select an organization and an SCM type. When you add a new project, the base path to the repository is given. Base path can be the link to your GitHub repository or the directory holding playbooks. If file is present on the AWX server, then select Manual and enter the base path to the file. In our case, we will select Git as our SCM Type because we will be using the GitHub repository. The other SCM types are Manual, Mercurial, Subversion, Red Hat Insights.*
5. *Enter SCM URL/Playbook directory. Here, we will add a public GitHub repository:* [*https://github.com/cupofcaffeine/ansible*](https://github.com/cupofcaffeine/ansible)*. You can select the required options from the SCM UPDATE OPTIONS section.*
6. *Click SAVE to save changes.*

*Figure 8. Creating a project*

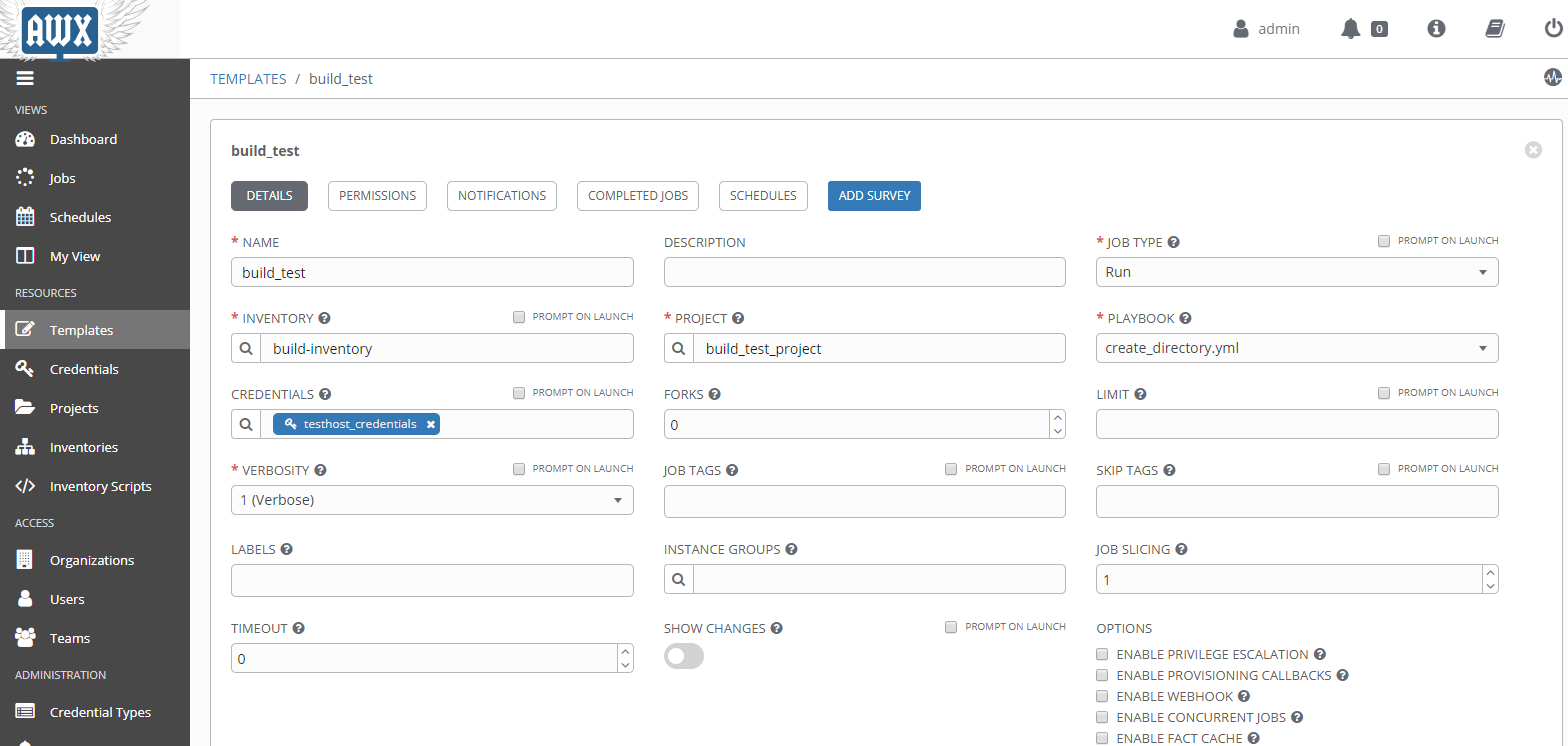
**

***8. Add a new template.***

*Here we select the specific playbook to be executed from the project we added.*

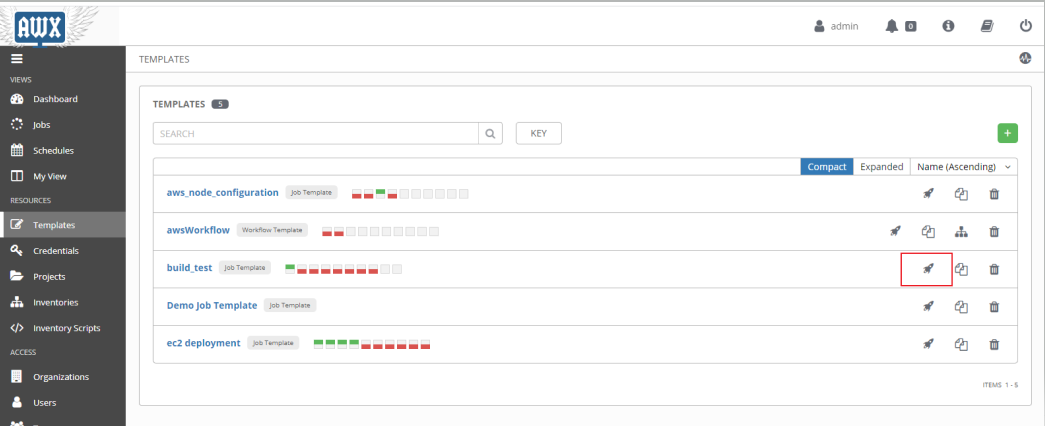
1. *On the left pane, click TEMPLATES.*
2. *Enter a name and description for the template.*
3. *Select the job type as Run or Check.*
4. *Select the inventory and the project in which your playbooks are present.*
5. *Select your playbook from the PLAYBOOK drop-down list.*
6. *Add credentials for the particular inventory of hosts.*
7. *Select your preference to log type in verbosity. Verbosity refers to the log type you might need while running the playbook.*
8. *Click SAVE to save changes.*

*Figure 9. Creating template*

**

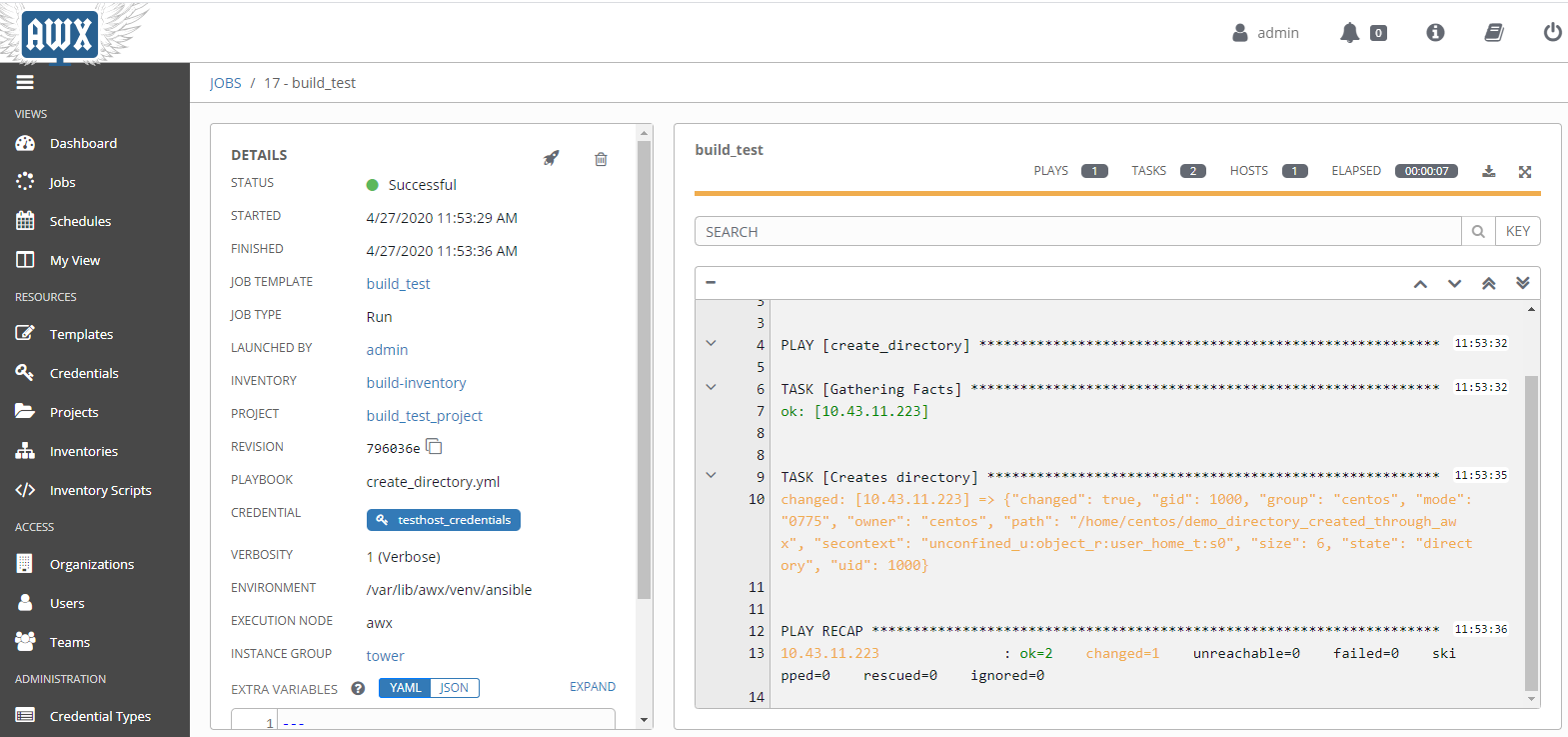
***9. Run the required job.***

1. *On the TEMPLATES page, select the template you want to run and click the job launcher icon.  
   Figure 10. Running a job*

**

1. *You will be redirected to the currently running job page. Notice that the verbose of the job you just ran is displayed.*
2. *Observe if the job ran successfully or not.*
   * *Green indicates that it is successful.*
   * *Orange indicates that the commands are executed, and changes are made/edited.*
   * *Red indicates warnings or errors.*

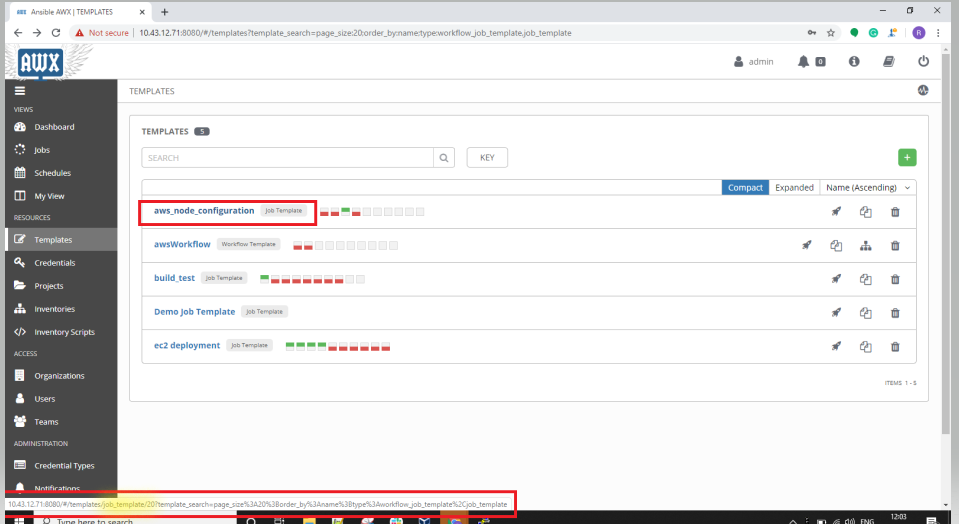
*Figure 11. Verbose to job executed*

**

***Ansible awx REST API access:***

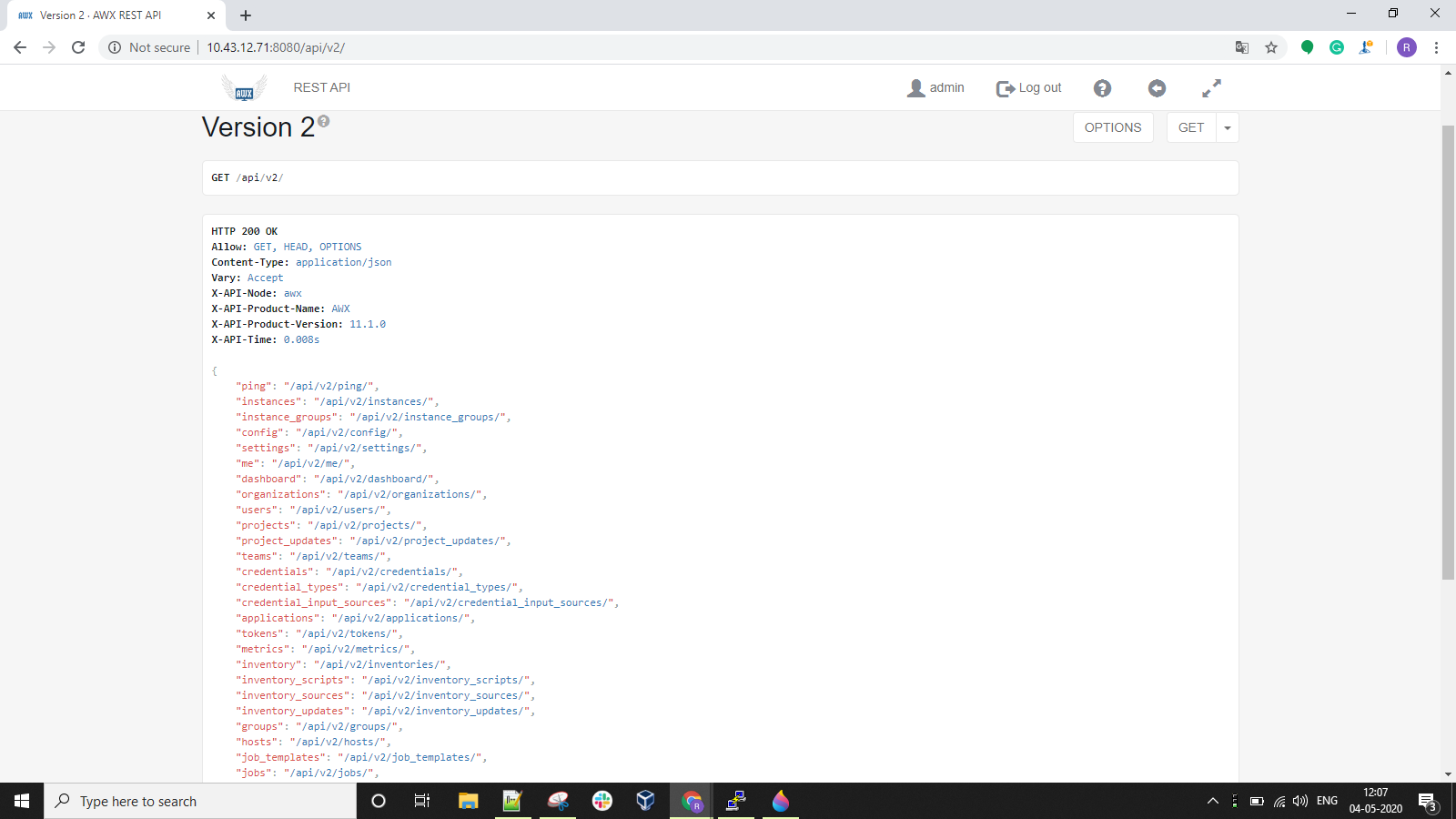
***Ansible AWX / Ansible Tower supports RESTfull API calls. It provides greater flexibility that you no need to be in Ansible Tower/AWX console to start the template or read the ansible job results. You could post the API from anywhere. It could be a web portal or from your laptop using POSTMAN/SoapUI. This article will demonstrate how to POST job using API and how to read the job result using the GET method.***

***1. Here is the template and template ID. If you keep the mouse near to that template, you can see the template id in the bottom of the screen.***

**

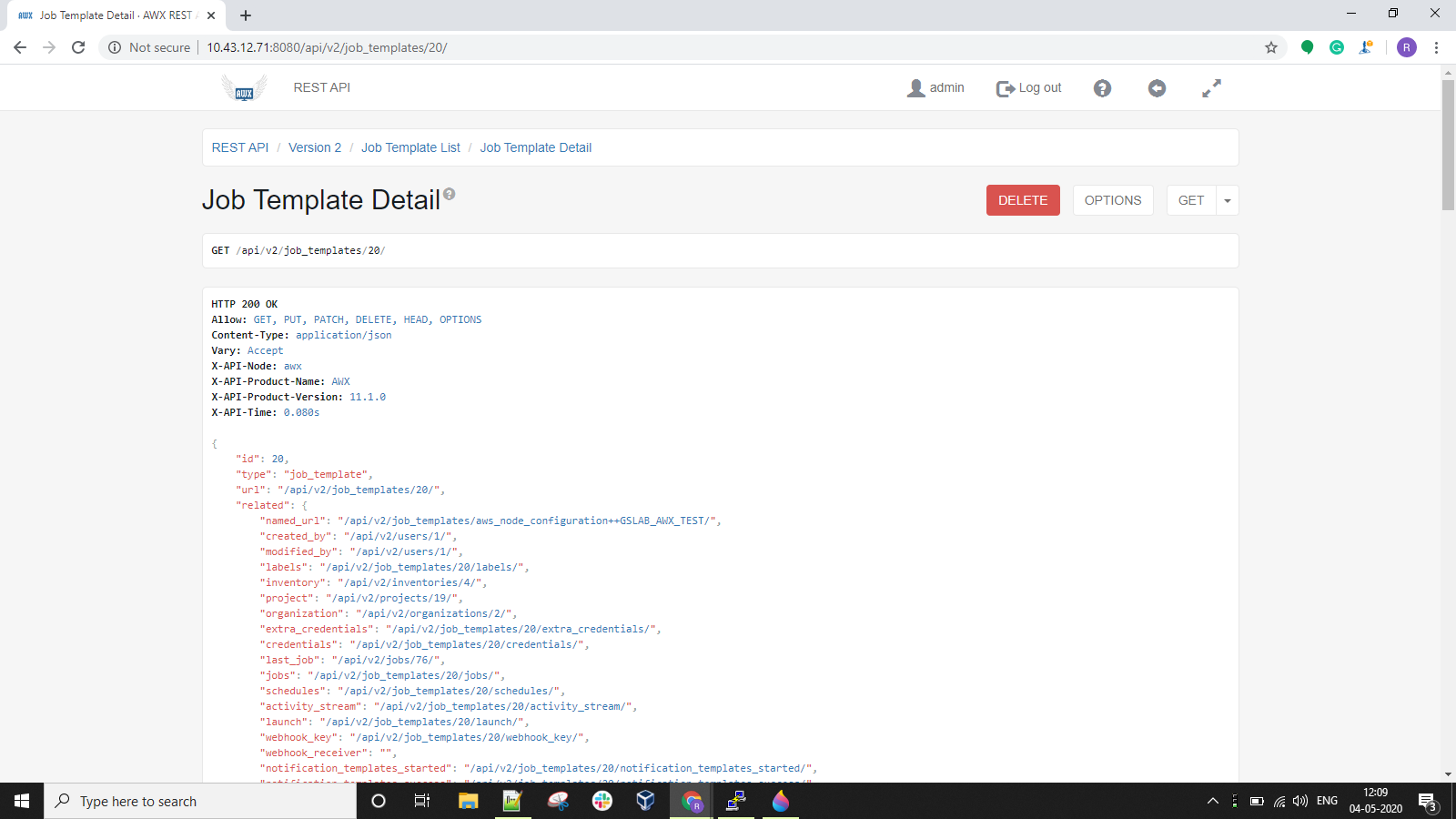
*2. How to navigate to AWX/ Ansible Tower API? Please refer to the following screenshot.*

[*http://10.43.12.71:8080/api/v2/*](http://10.43.12.71:8080/api/v2/)

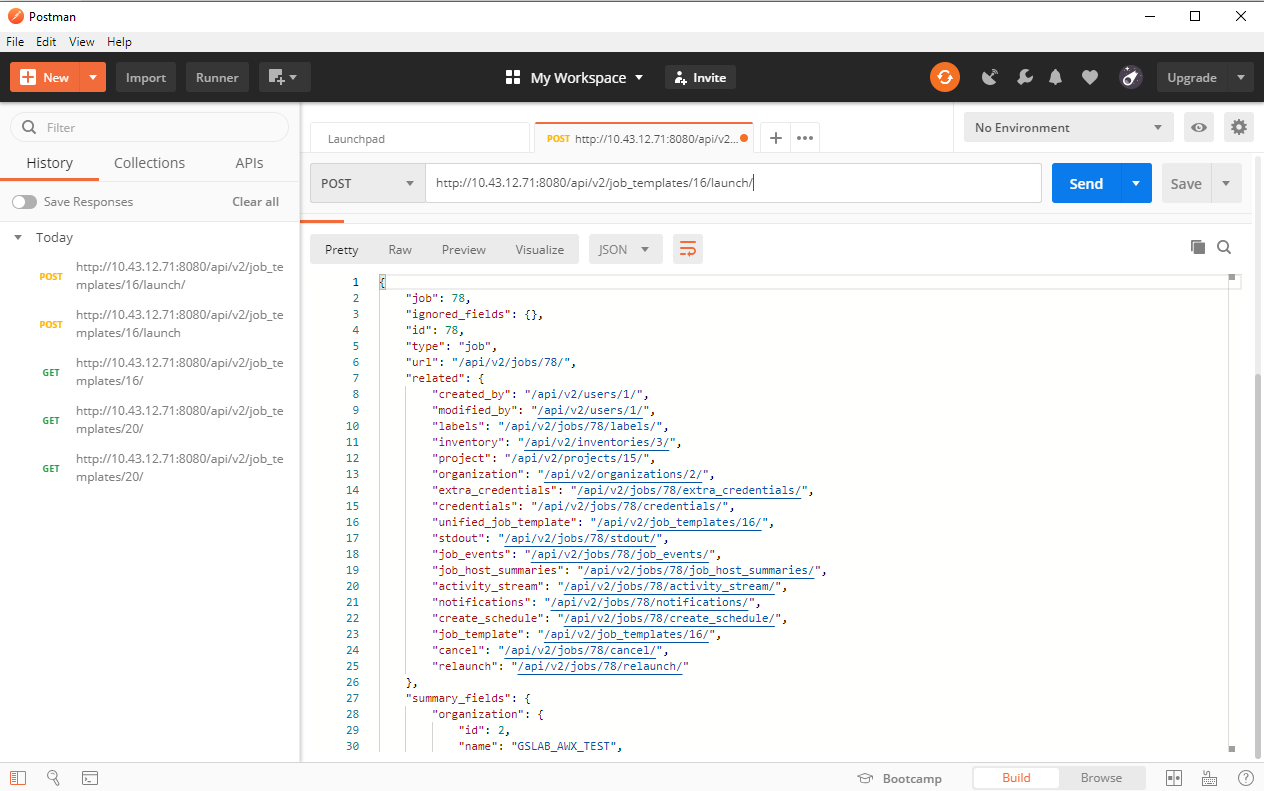
**

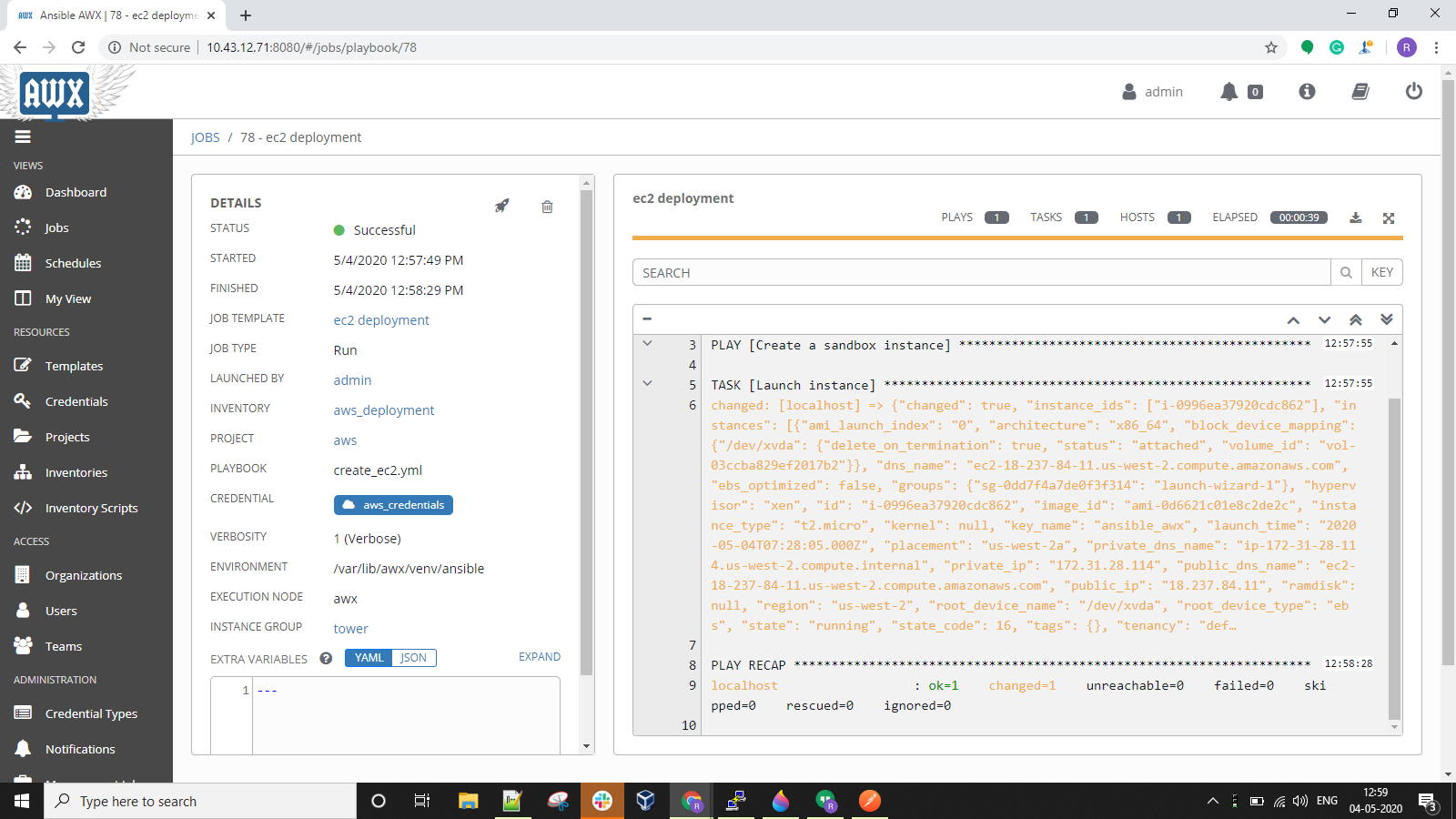
*3. You could access the job template REST API like below.*

*For Ex.:* [*http://10.43.12.71:8080/api/v2/job\_templates/20/*](http://10.43.12.71:8080/api/v2/job_templates/20/)

**

*4. Launched an template using postman app*

**

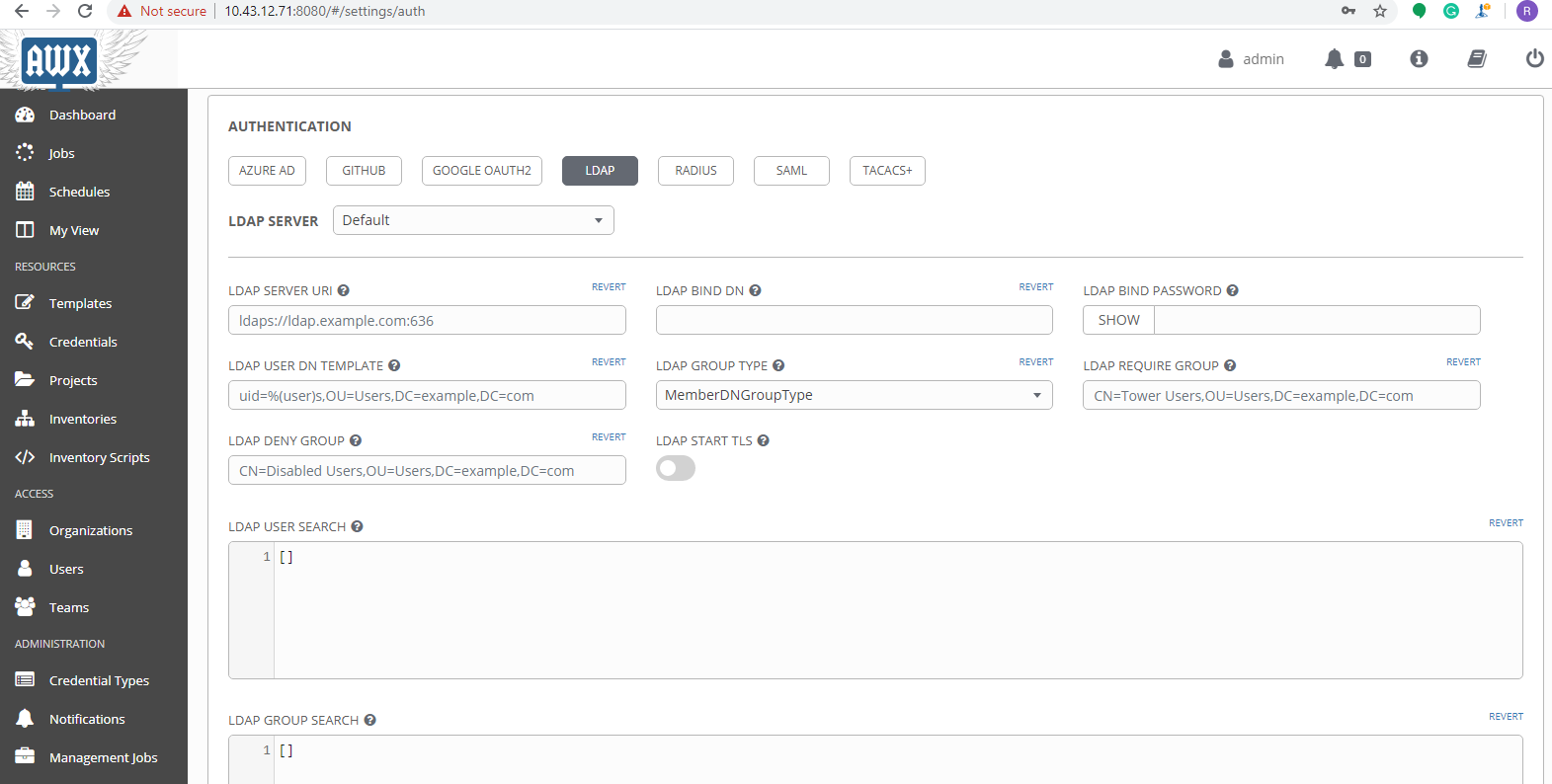


**LDAP configuration:**

**Login with admin credentials and configure LDAP**

Go to **setting** → **AUTHENTICATION → Choose LDAP and fill required details for configuration.**

[**https://docs.ansible.com/ansible-tower/latest/html/administration/ldap\_auth.html**](https://docs.ansible.com/ansible-tower/latest/html/administration/ldap_auth.html)



**Project hierarchy:**

**User 1, User2, User3 User 1, User2, User3**