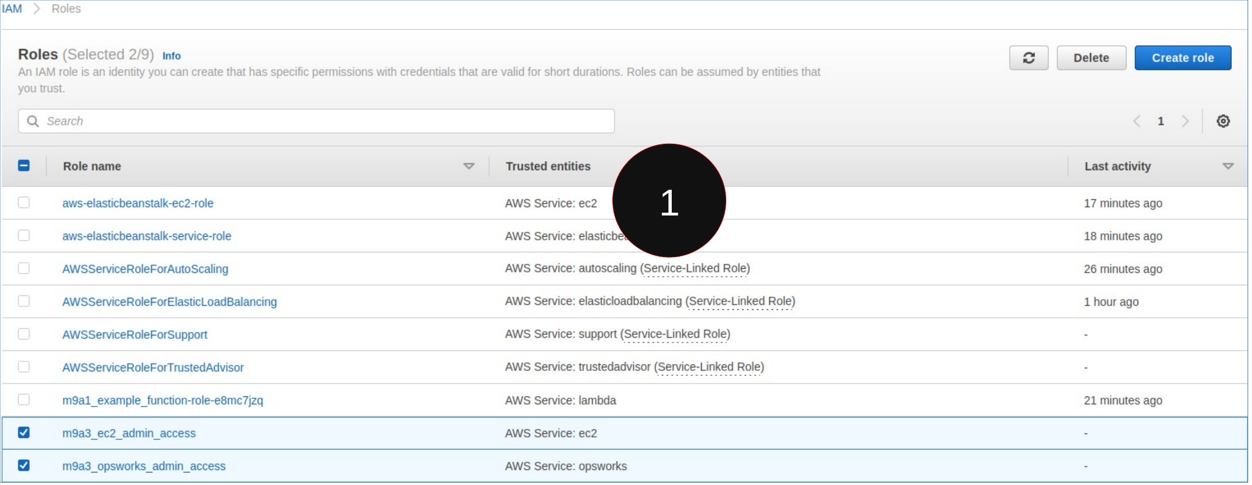
Module 9 Assignment 3: Elastic BeanStalk

You have been asked to:

1. Create an OpsWorks sample stack, start the instances and deploy the application
2. Add 2 more t2.medium instances
3. Make a change to the repository code and check if it reflects in all the instances



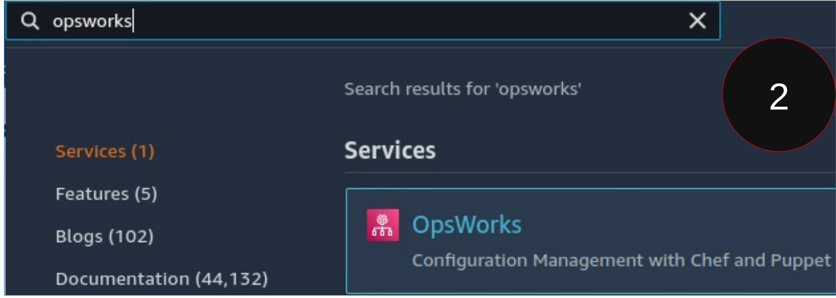
Roles (Selected 2/9) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Delete | Create role

Search

	Role name	Trusted entities	Last activity
□	aws-elasticbeanstalk-ec2-role	AWS Service: ec2	17 minutes ago
□	aws-elasticbeanstalk-service-role	AWS Service: elasticbe	18 minutes ago
□	AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Service-Linked Role)	26 minutes ago
□	AWSServiceRoleForElasticLoadBalancing	AWS Service: elasticloadbalancing (Service-Linked Role)	1 hour ago
□	AWSServiceRoleForSupport	AWS Service: support (Service-Linked Role)	-
□	AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linked Role)	-
□	m9a1_example_function-role-e8mc7jzq	AWS Service: lambda	21 minutes ago
☑	m9a3_ec2_admin_access	AWS Service: ec2	-
☑	m9a3_opsworks_admin_access	AWS Service: opsworks	-



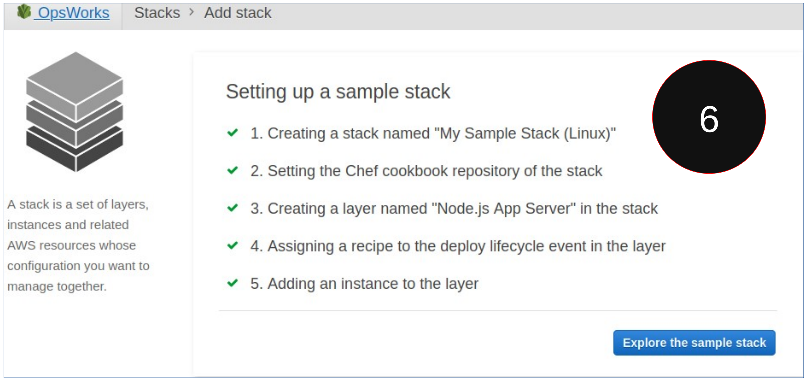
opsworks

Search results for 'opsworks'

Services (1)
Features (5)
Blogs (102)
Documentation (44,132)

Services

OpsWorks
Configuration Management with Chef and Puppet



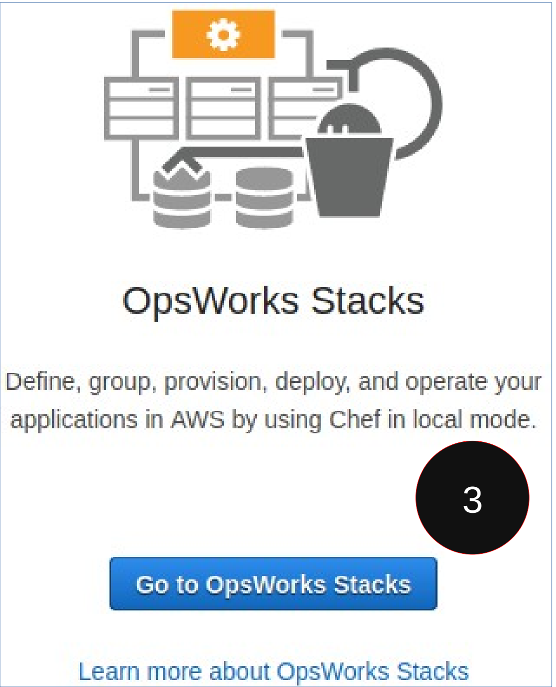
OpsWorks > Stacks > Add stack

A stack is a set of layers, instances and related AWS resources whose configuration you want to manage together.

Setting up a sample stack

- ✓ 1. Creating a stack named "My Sample Stack (Linux)"
- ✓ 2. Setting the Chef cookbook repository of the stack
- ✓ 3. Creating a layer named "Node.js App Server" in the stack
- ✓ 4. Assigning a recipe to the deploy lifecycle event in the layer
- ✓ 5. Adding an instance to the layer

Explore the sample stack

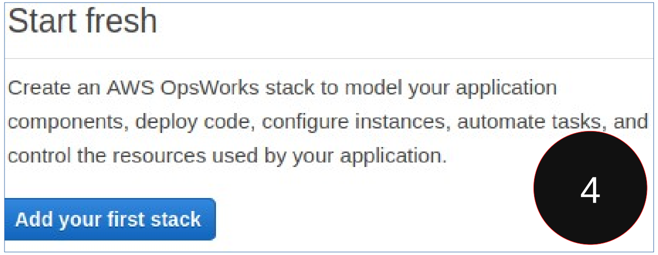


OpsWorks Stacks

Define, group, provision, deploy, and operate your applications in AWS by using Chef in local mode.

Go to OpsWorks Stacks

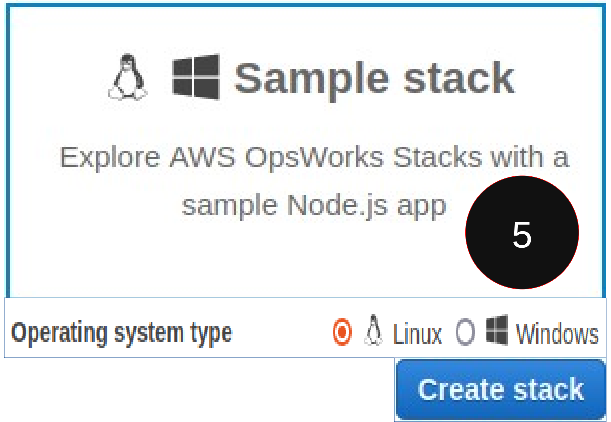
Learn more about OpsWorks Stacks



Start fresh

Create an AWS OpsWorks stack to model your application components, deploy code, configure instances, automate tasks, and control the resources used by your application.

Add your first stack



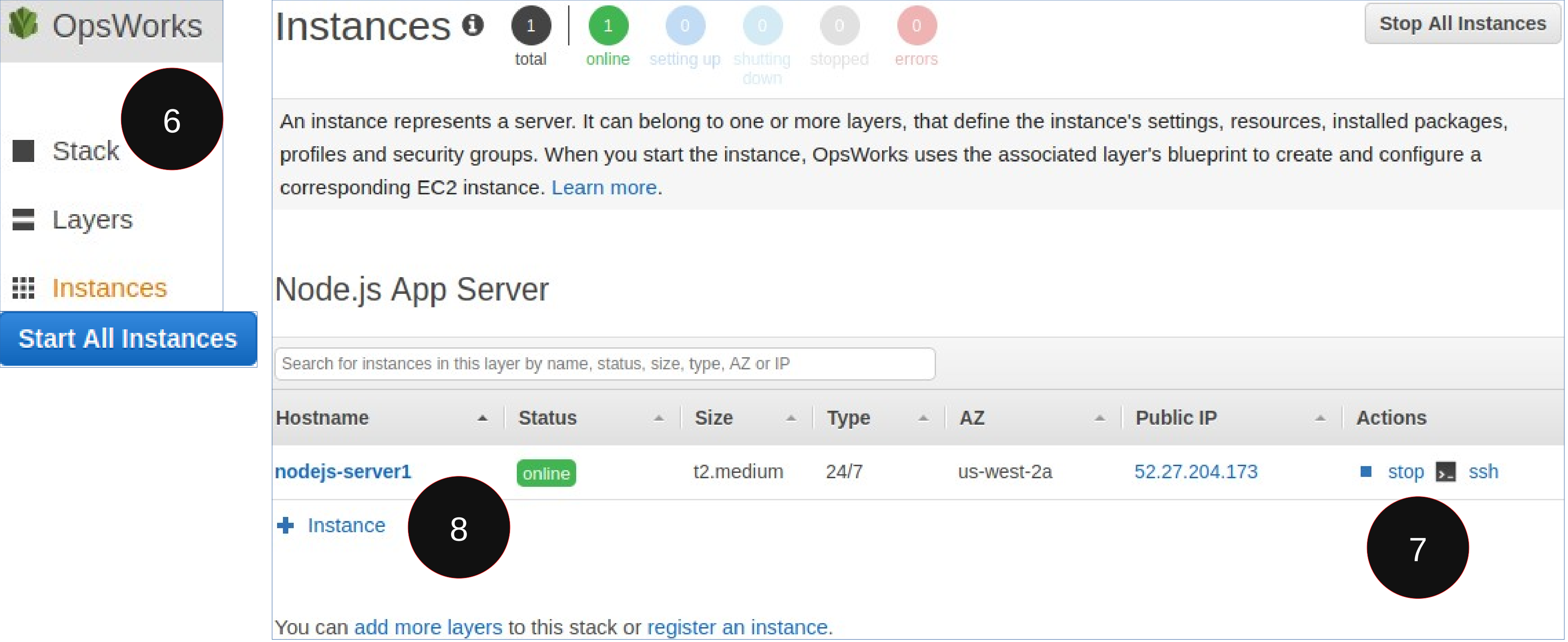
Sample stack

Explore AWS OpsWorks Stacks with a sample Node.js app

Operating system type ● Linux ○ Windows

Create stack

1. Open IAM Console. Create roles to provide EC2 admin access and OpsWorks admin access
2. Open AWS Console. Search for “Opsworks”. Open AWS OpsWorks console
3. Select “OpsWorks Stacks”
4. Select “Add your first stack”
5. Select “sample stack”, select “Linux” as operating system type. Click on “Create stack”. Ensure that the IAM role has rights to create AWS OpsWorks stack. Else, create an IAM role for this
6. Wait till stack is created



An instance represents a server. It can belong to one or more layers, that define the instance's settings, resources, installed packages, profiles and security groups. When you start the instance, OpsWorks uses the associated layer's blueprint to create and configure a corresponding EC2 instance. Learn more.

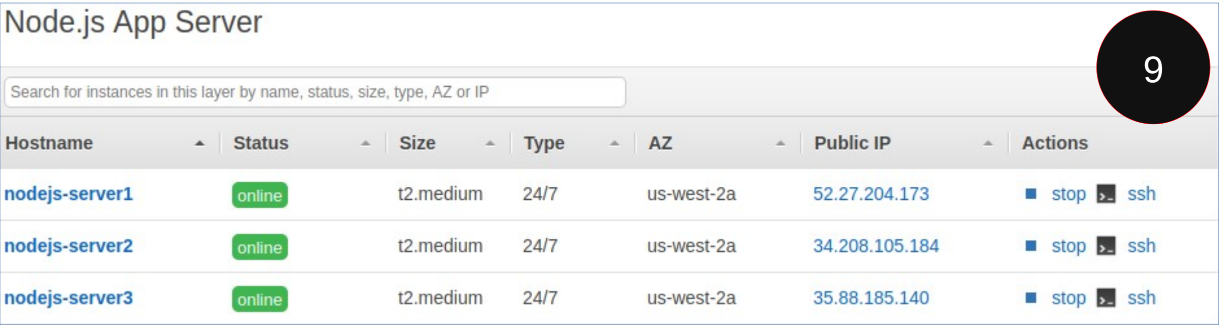
Node.js App Server

Search for instances in this layer by name, status, size, type, AZ or IP

Hostname	Status	Size	Type	AZ	Public IP	Actions
nodejs-server1	online	t2.medium	24/7	us-west-2a	52.27.204.173	■ stop ssh

✚ Instance

You can add more layers to this stack or register an instance.



Node.js App Server

Search for instances in this layer by name, status, size, type, AZ or IP

Hostname	Status	Size	Type	AZ	Public IP	Actions
nodejs-server1	online	t2.medium	24/7	us-west-2a	52.27.204.173	■ stop ssh
nodejs-server2	online	t2.medium	24/7	us-west-2a	34.208.105.184	■ stop ssh
nodejs-server3	online	t2.medium	24/7	us-west-2a	35.88.185.140	■ stop ssh

6. Open “Instances” console. Start all instances
7. Start the app instance and verify that it is successfully started
8. Click on “instance” to create 2 more “t2.medium” instances
9. Start all instances and verify that they are started correctly. Click on the public IP of each instance and verify that we are able to see the app frontend UI in the browser