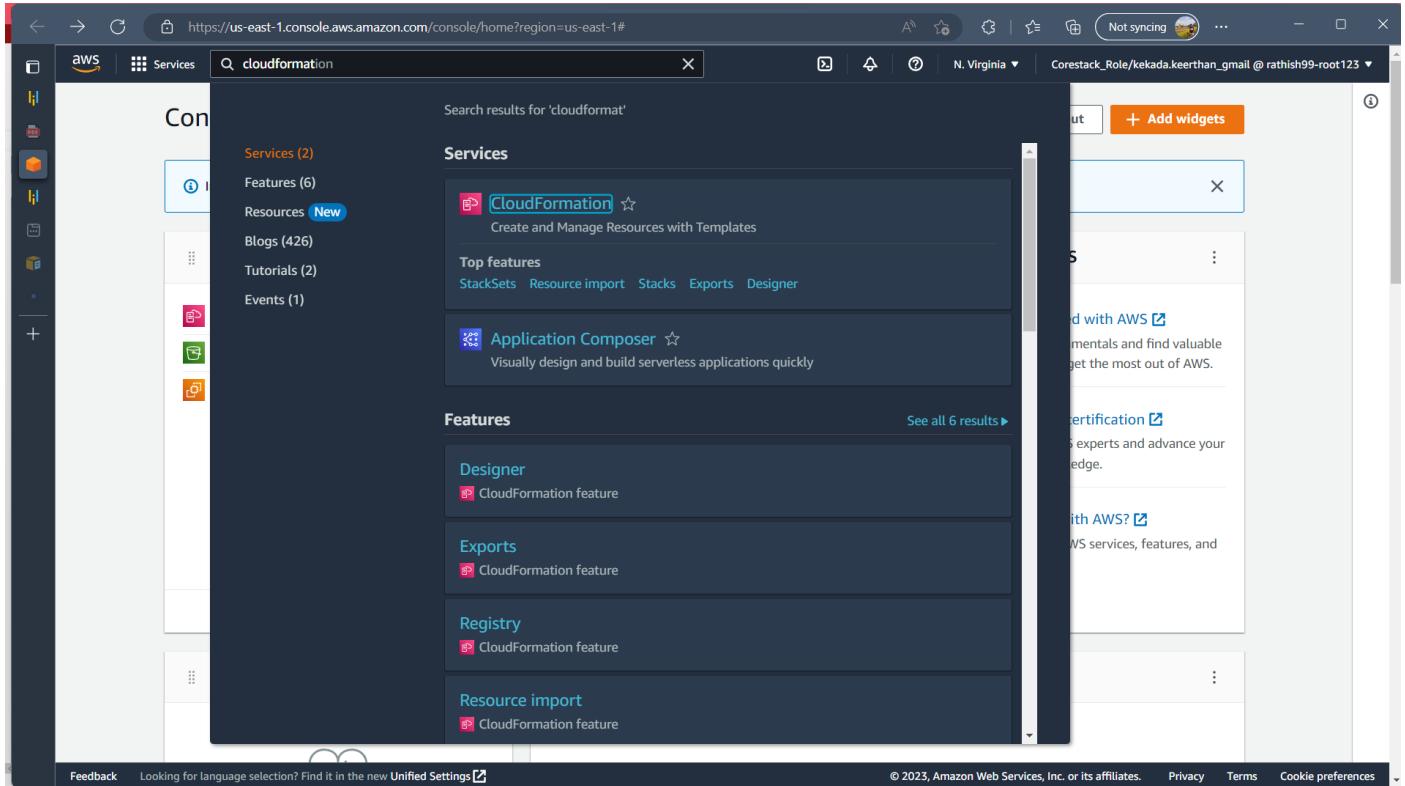


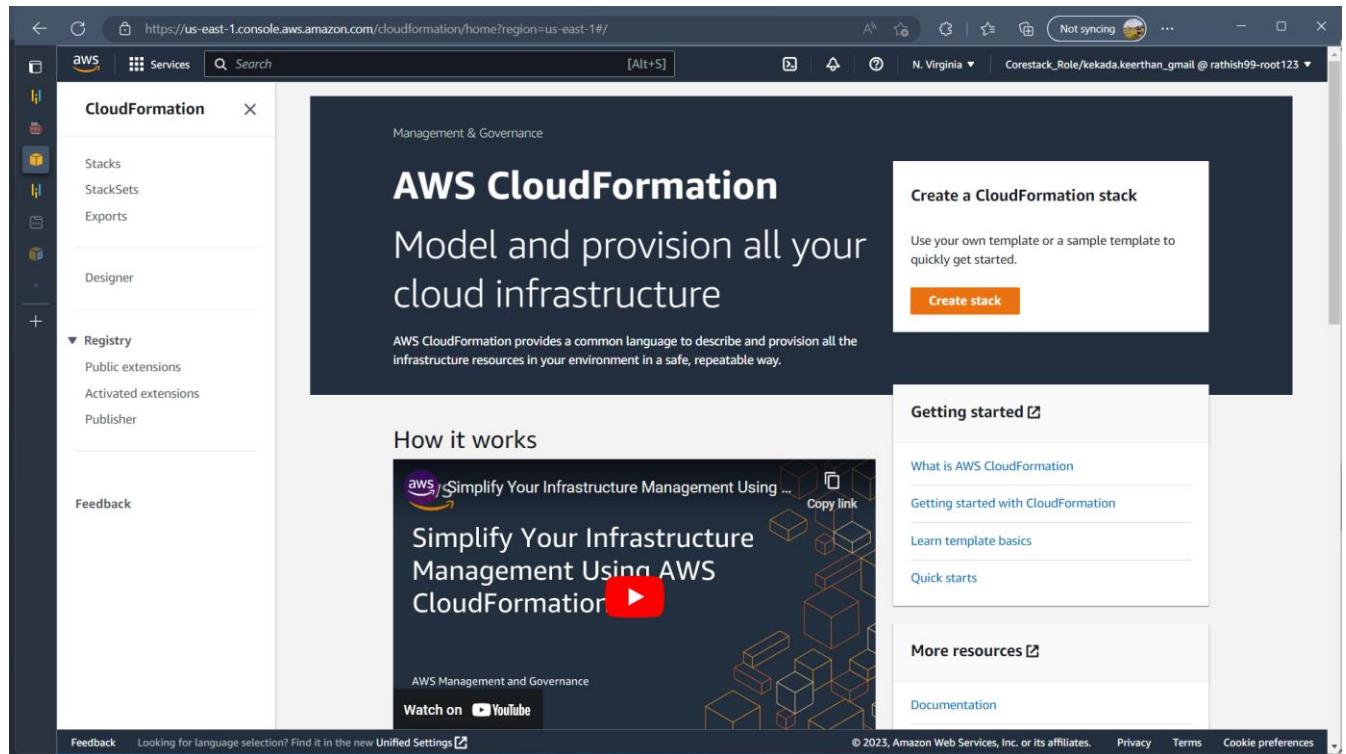
A}. Create a CloudFormation stack

1. Open AWS console and search for cloudformation and click on it.



The screenshot shows the AWS console search results for 'cloudformat'. The 'CloudFormation' service is highlighted with a blue box. The search bar at the top contains 'cloudformat'. The 'Services' section lists 'CloudFormation' under 'Services (2)' and other services like 'Application Composer'. The 'Features' section lists 'Designer', 'Exports', 'Registry', and 'Resource import' as CloudFormation features. A sidebar on the right provides information about CloudFormation, such as 'Get started with AWS' and 'Certification'.

2. Click on create stack .



The screenshot shows the AWS CloudFormation home page. The left sidebar includes options like 'Stacks', 'StackSets', 'Exports', 'Designer', 'Registry', 'Feedback', and 'Public extensions'. The main content area features the 'AWS CloudFormation' logo and the tagline 'Model and provision all your cloud infrastructure'. It explains that CloudFormation provides a common language to describe and provision infrastructure resources. A large 'Create a CloudFormation stack' button is prominently displayed. Below it, there's a 'How it works' section with a video thumbnail titled 'Simplify Your Infrastructure Management Using AWS CloudFormation' and a 'Getting started' sidebar with links to 'What is AWS CloudFormation', 'Getting started with CloudFormation', 'Learn template basics', and 'Quick starts'.

3.Click on use a sample template and choose wordpress blog template.

The screenshot shows the AWS CloudFormation 'Create stack' wizard. On the left, a sidebar lists steps: Step 1 (Create stack), Step 2 (Specify stack details), Step 3 (Configure stack options), and Step 4 (Review). The main area is titled 'Prerequisite - Prepare template'. It contains a section 'Prepare template' with three options: 'Template is ready' (unchecked), 'Use a sample template' (checked), and 'Create template in Designer' (unchecked). Below this is a 'Select a sample template' section. A dropdown menu shows 'WordPress blog' as the selected item. To the right of the dropdown is a link 'View more sample templates'. At the bottom right are 'Cancel' and 'Next' buttons.

This screenshot shows the same 'Create stack' wizard as the previous one, but with a search bar at the top containing the text 'Simple'. The search results for 'Simple' include 'LAMP Stack', 'Ruby on Rails Stack', and 'WordPress blog'. The 'WordPress blog' entry is highlighted. The rest of the interface is identical to the first screenshot, showing the 'Use a sample template' option selected and the 'WordPress blog' template chosen in the dropdown.

4. Give stack name as `wordpress-server1` and type in the password of your choice.

Step 2
Specify stack details

Stack name

Stack name
`wordpress-server1`

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

DBName
The WordPress database name
`wordpressdb`

DBPassword
The WordPress database admin account password
`*****`

DBRootPassword
MySQL root password
`*****`

DBUser
The WordPress database admin account username
`*****`

InstanceType
WebServer EC2 instance type
`t2.micro`

5. Select `t2.micro` free tier for instance type.

CloudFormation

Stacks

StackSets

Exports

Designer

Registry

Public extensions

Activated extensions

Publisher

Feedback

DBName
The WordPress database name
`wordpressdb`

DBPassword
The WordPress database admin account password
`*****`

DBRootPassword
MySQL root password
`*****`

DBUser
The WordPress database admin account username
`*****`

InstanceType
WebServer EC2 instance type
`t2.micro`

KeyName
Name of an existing EC2 KeyPair to enable SSH access to the instances
`Wordpress key`

SSHLocation
The IP address range that can be used to SSH to the EC2 instances
`0.0.0.0/0`

Cancel Previous Next

6. Select the created key pair and click next.

The screenshot shows the AWS CloudFormation 'Create New Stack' wizard, Step 3: Set Parameters. The left sidebar lists 'CloudFormation' services: Stacks, StackSets, Exports, Designer, Registry (Public extensions, Activated extensions, Publisher), and Feedback. The main panel contains the following parameters:

- DBName**: The WordPress database name. Value: wordpressdb.
- DBPassword**: The WordPress database admin account password. Value: [REDACTED]
- DBRootPassword**: MySQL root password. Value: [REDACTED]
- DBUser**: The WordPress database admin account username. Value: [REDACTED]
- InstanceType**: WebServer EC2 instance type. Value: t2.micro.
- KeyName**: Name of an existing EC2 KeyPair to enable SSH access to the instances. Value: Wordpress key.
- SSHLocation**: The IP address range that can be used to SSH to the EC2 instances. Value: 0.0.0.0/0.

At the bottom right are buttons: Cancel, Previous, and Next (highlighted in orange).

7. Keep default settings for 'configure stack options' and click next.

The screenshot shows the AWS CloudFormation 'Create New Stack' wizard, Step 4: Configure stack options. The left sidebar lists 'CloudFormation' services: Stacks, StackSets, Exports, Designer, Registry (Public extensions, Activated extensions, Publisher), and Feedback. The main panel has two sections:

- Behavior on provisioning failure**: Specifying the roll back behavior for a stack failure. Options:
 - Roll back all stack resources: Roll back the stack to the last known stable state.
 - Preserve successfully provisioned resources: Preserves the state of successfully provisioned resources, while rolling back failed resources to the last known stable state. Resources without a last known stable state will be deleted upon the next stack operation.
- Advanced options**: You can set additional options for your stack, like notification options and a stack policy.
 - Stack policy**: Defines the resources that you want to protect from unintentional updates during a stack update.
 - Rollback configuration**: Specify alarms for CloudFormation to monitor when creating and updating the stack. If the operation breaches an alarm threshold, CloudFormation rolls it back.
 - Notification options**
 - Stack creation options**

At the bottom right are buttons: Cancel, Previous, and Next (highlighted in orange).

8. Review and click submit.

The screenshot shows the AWS CloudFormation console interface. On the left, the navigation bar includes 'Services' and a search bar. The main area displays a list of stacks under 'CloudFormation > Stacks'. One stack, 'wordpress-server1', is selected and highlighted with a blue border. The stack details are shown on the right, including its Stack ID (arn:aws:cloudformation:us-east-1:1545241677902:stack/wordpress-server1/d3382980-9557-11ed-84ef-0e8a9b07c009), a detailed description of the AWS CloudFormation Sample Template, and its current status as 'CREATE_IN_PROGRESS'. Other stack details like Root stack, Created time, and Updated time are also listed.

9. Go to output tab after the creation of stack and copy the URL and paste in new tab of your browser.

The screenshot shows the 'Events' tab for the 'wordpress-server1' stack. The tab header indicates there are 9 events. The table below lists the events, showing the timestamp, logical ID, status, and status reason. The first two events are for the 'wordpress-server1' resource, both marked as 'CREATE_COMPLETE'. The third event is for the 'WebServer' resource, also marked as 'CREATE_COMPLETE' with a status reason indicating it received a SUCCESS signal with UniqueId i-0b12d9067649e015a. Subsequent events are for the 'WebServer' and 'WebServerSecurityGroup' resources, all in 'CREATE_IN_PROGRESS' status.

Timestamp	Logical ID	Status	Status reason
2023-01-16 10:11:47 UTC+0530	wordpress-server1	CREATE_COMPLETE	-
2023-01-16 10:11:45 UTC+0530	WebServer	CREATE_COMPLETE	-
2023-01-16 10:11:44 UTC+0530	WebServer	CREATE_IN_PROGRESS	Received SUCCESS signal with UniqueId i-0b12d9067649e015a
2023-01-16 10:10:11 UTC+0530	WebServer	CREATE_IN_PROGRESS	Resource creation Initiated
2023-01-16 10:10:09 UTC+0530	WebServer	CREATE_IN_PROGRESS	-
2023-01-16 10:10:07 UTC+0530	WebServerSecurityGroup	CREATE_COMPLETE	-
2023-01-16 10:10:06 UTC+0530	WebServerSecurityGroup	CREATE_IN_PROGRESS	-
2023-01-16 10:10:05 UTC+0530	WebServerSecurityGroup	CREATE_IN_PROGRESS	-

CloudFormation > Stacks > wordpress-server1

Stacks (1)

Active

Stacks

wordpress-server1
2023-01-16 10:09:55 UTC+0530
CREATE_COMPLETE

wordpress-server1

Delete Update Stack actions Create stack

Stack info Events Resources Outputs Parameters Template Change sets

Resources (2)

Logical ID	Physical ID	Type	Status	Module
WebServer	i-0b12d9067649e015a	AWS::EC2::Instance	CREATE_COMPLETE	
WebServerSecurityGroup	wpserver1-WebServerSecurityGroup-YQU78ZUIEXQU	AWS::EC2::SecurityGroup	CREATE_COMPLETE	

Feedback Looking for language selection? Find it in the new Unified Settings © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

CloudFormation > Stacks > wordpress-server1

Stacks (1)

Active

Stacks

wordpress-server1
2023-01-16 10:09:55 UTC+0530
CREATE_COMPLETE

wordpress-server1

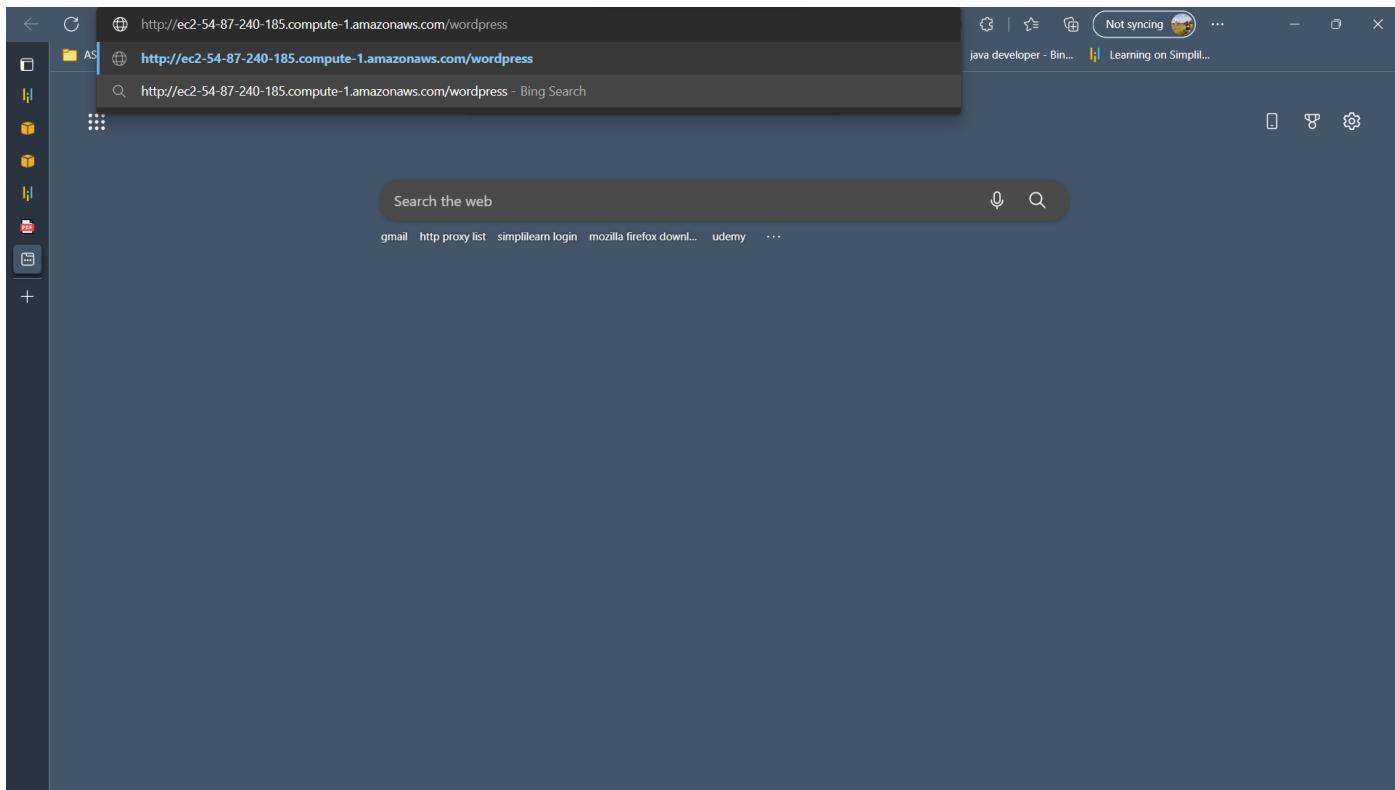
Delete Update Stack actions Create stack

Stack info Events Resources Outputs Parameters Template Change sets

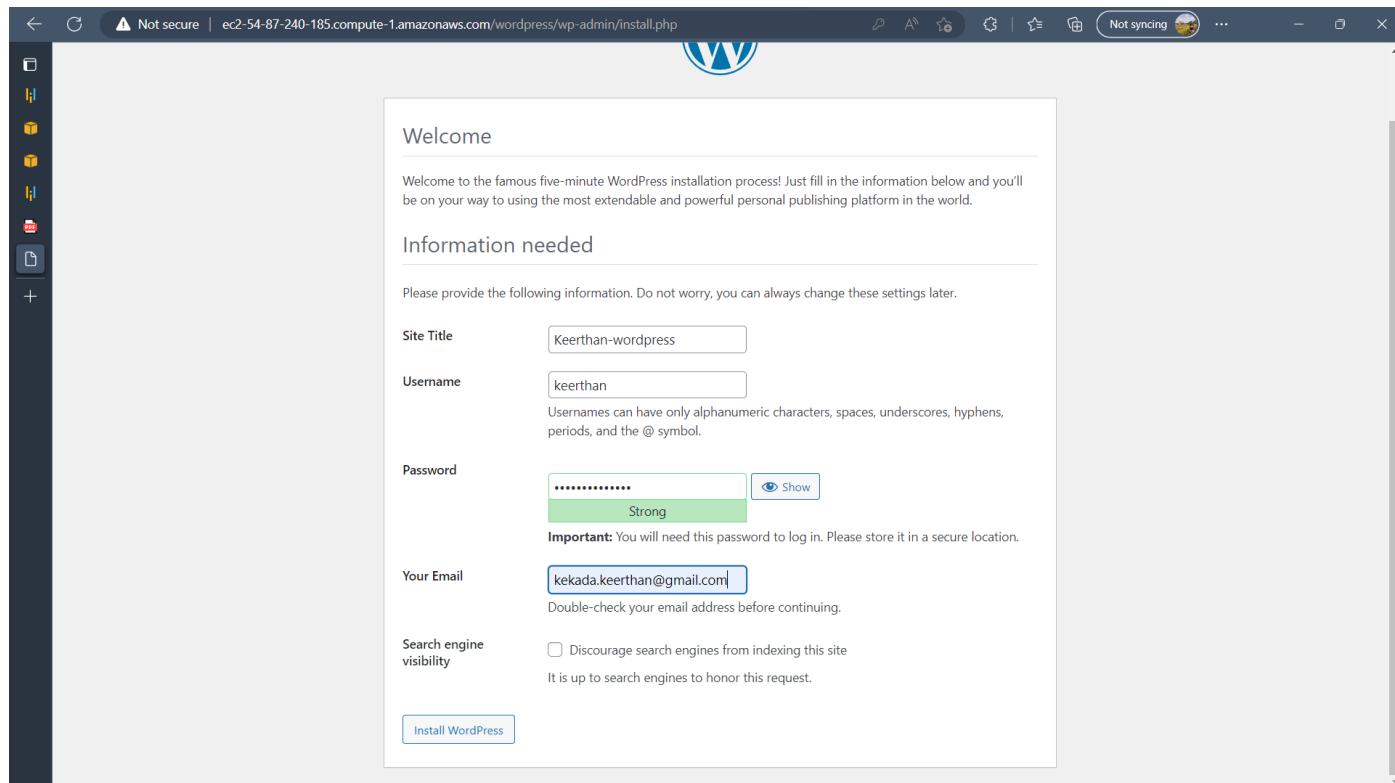
Outputs (1)

Key	Value	Description	Export name
WebsiteURL	http://ec2-54-87-240-185.compute-1.amazonaws.com/wordpress	WordPress Website	-

Feedback Looking for language selection? Find it in the new Unified Settings © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



10.Fill up the details and click install wordpress.



The screenshot shows the WordPress dashboard. On the left, there's a sidebar with navigation links: Home, Updates (1), Posts, Media, Pages, Comments, Appearance, Plugins (1), Users, Tools, Settings, and a 'Collapse menu' option. The main area features a large blue banner with the text "Welcome to WordPress!" and a link to "Learn more about the 6.1.1 version". Below the banner, there are three cards: one for authoring content with blocks and patterns, another for customizing the site with block themes, and a third for switching up the site's look & feel with styles. At the bottom left, a notification box says "PHP Update Recommended" with a message about outdated PHP version 7.3.30. To the right, there's a "Quick Draft" section with fields for Title and Content.

11.Login to your wordpress

The screenshot shows the WordPress login page. It features the classic blue 'W' logo at the top. Below it is a form with fields for "Username or Email Address" (containing "keerthan") and "Password" (represented by a series of dots). There are checkboxes for "Remember Me" and a "Log In" button. Below the form, there are links for "Lost your password?" and "← Go to Keerthan-wordpress".

B}.Create an AMI of the WordPress instance

1.Go to search option and search for EC2.

2.Go to the instances and select the instance which was created using cloudformation.

The screenshot shows the AWS EC2 Instances page. The URL in the browser is <https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances>. The page displays one instance:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public I
wordpress server	i-0b12d9067649e015a	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-54-

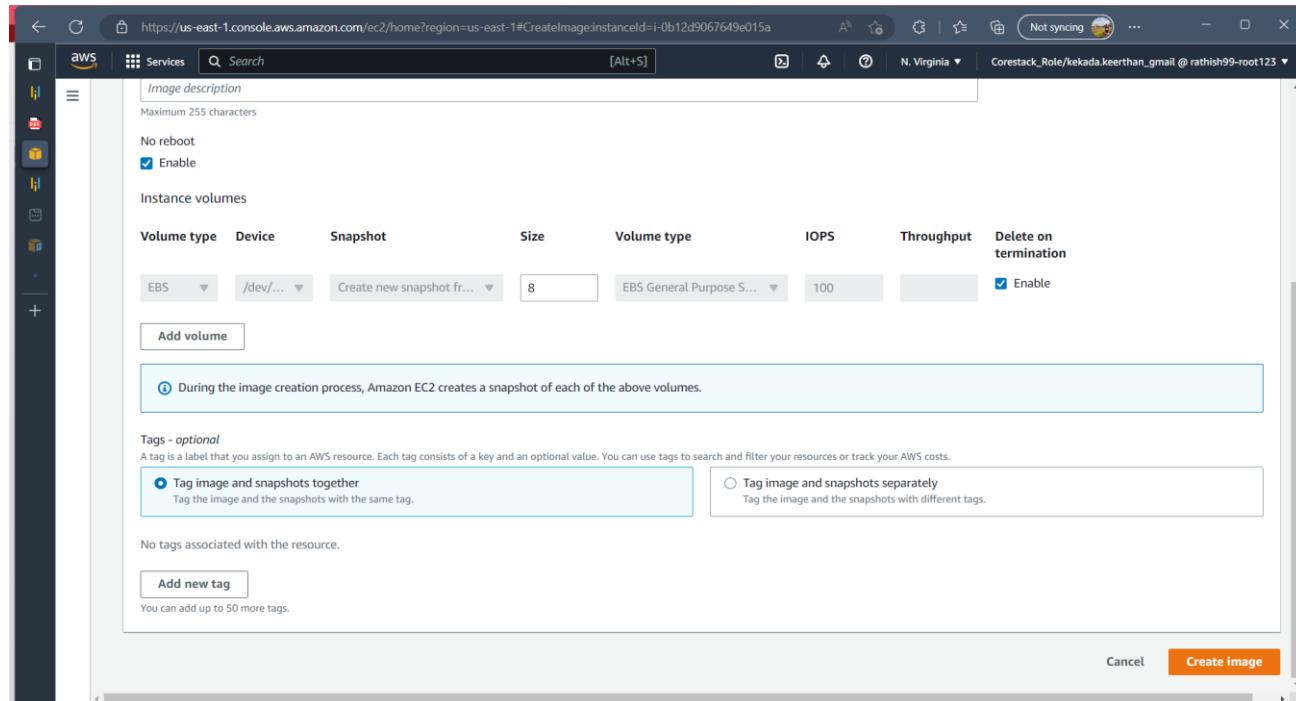
A modal window titled "Select an instance" is open at the bottom of the screen, indicating the user is about to interact with the selected instance.

3.Click on actions and select ‘Images and template’.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various navigation options like EC2 Dashboard, EC2 Global View, Events, Tags, Limits, Instances (selected), Images, AMIs, and Elastic Block Store. The main area displays a table with one instance: Name (wordpress server), Instance ID (i-0b12d9067649e015a), Instance state (Running), Instance type (t2.micro), and Status check (2/2 checks passed). Below the table, there's an 'Instance summary' section with details like Public IPv4 address (54.87.240.185), Instance state (Running), and Instance type (t2.micro). The Actions menu on the right is open, and the 'Image and templates' option is highlighted with a blue border.

This screenshot is identical to the one above, showing the AWS EC2 Instances page with the same instance details and Actions menu. The 'Image and templates' option in the Actions menu is again highlighted with a blue border.

4.Select create image and give details then click create image.



5. AMI

is created.

The screenshot shows the 'Amazon Machine Images (AMIs)' page in the AWS EC2 console. A single AMI is listed:

Name	AMI ID	AMI name	Source	Owner	Visibility
Wordpress	ami-009daa9170a422aae	Wordpress ami	S45241677902/Wordpress ami	545241677902	Private

A detailed view of the AMI is shown in a modal window:

AMI ID: ami-009daa9170a422aae (Wordpress)

Details			
AMI ID	ami-009daa9170a422aae (Wordpress)	Image type	machine
AMI name	Wordpress ami	Owner account ID	S45241677902
Root device name	/dev/xvda	Status	Pending
Boot mode	-	State reason	-
Creation date	Mon Jan 16 2023 10:25:54 GMT+0530 (India Standard Time)	Kernel ID	-
Platform details	Linux/UNIX	Root device type	EBS
Architecture	x86_64	Usage operation	RunInstances
Source	S45241677902/Wordpress ami	Virtualization type	hvm

C}.Configure Auto Scaling to launch a new WordPress instance

1.Goto launch template and click on create launch template.

The screenshot shows the AWS EC2 Launch Templates page. On the left, there's a sidebar with navigation links like EC2 Dashboard, EC2 Global View, Events, Tags, Limits, Instances, Launch Templates (which is selected), Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations, Images, AMIs, and Elastic Block Store. The main content area has a title 'EC2 launch templates' and sub-section 'Streamline, simplify and standardize instance launches'. It includes a description of launch templates and a 'New launch template' button with an orange 'Create launch template' button below it. To the right, there's a 'Documentation' section with links to Documentation and API reference.

2.Give a name and select ‘my AMI’s’ in Application and OS Images (Amazon Machine Image) section.

The screenshot shows the 'Create Template' wizard. In the 'Launch template name - required' field, 'Wordpress-Autoscaling' is entered. In the 'Template version description' field, 'A prod webserver for MyApp' is entered. Under 'Auto Scaling guidance', there's a checkbox for 'Provide guidance to help me set up a template that I can use with EC2 Auto Scaling'. Below these, there are sections for 'Template tags' and 'Source template'. The 'Launch template contents' section is collapsed. The 'Application and OS Images (Amazon Machine Image)' section is expanded, showing a search bar with 'Search our full catalog including 1000s of application and OS images' and tabs for 'Recents', 'My AMIs', and 'Quick Start'. A tooltip for the 'Free tier' is visible, stating: 'Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GB of EBS storage, 2 million IOPS, 1 GB of snapshots, and 100 GB of bandwidth to the internet.' At the bottom right, there are 'Cancel' and 'Create launch template' buttons.

The screenshot shows the AWS CloudFormation 'Create template' wizard. The current step is 'Application and OS Images (Amazon Machine Image)'. The left panel displays a search bar and filters for 'Template tags' and 'Source template'. The main content area shows the 'Launch template contents' section, which includes a search bar for AMIs, a filter for 'Owned by me', and a list of available AMIs. One item, 'Wordpress ami', is selected. The right panel contains a 'Summary' section with details like 'Software Image (AMI)', 'Virtual server type (instance type)', 'Firewall (security group)', and 'Storage (volumes)'. A callout box highlights the 'Free tier' information: 'In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.' At the bottom right are 'Cancel' and 'Create launch template' buttons.

3. Select instance type as t2.micro.

The screenshot shows the AWS CloudFormation 'Create template' wizard. The current step is 'Instance type'. The left panel displays a search bar and filters for 'Template tags' and 'Source template'. The main content area shows the 'Launch template contents' section, specifically the 'Instance type' configuration. It includes a dropdown menu for selecting an instance type, currently set to 'Don't include in launch template'. Below the dropdown, a list of available instance types is shown: t1.micro, t2.nano, t2.micro, and t2.small. Each item provides a brief description of its family, vCPU count, memory, and pricing. The right panel contains a 'Summary' section with details like 'Software Image (AMI)', 'Virtual server type (instance type)', 'Firewall (security group)', and 'Storage (volumes)'. A callout box highlights the 'Free tier' information: 'In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.' At the bottom right are 'Cancel' and 'Create launch template' buttons.

4. Select the key pair which was created initially.

The screenshot shows the AWS EC2 console with the URL <https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#CreateTemplate>. The page is titled 'Create template' and is on step 2 of 3. The 'Key pair (login)' section is active, showing a dropdown menu for 'Key pair name'. The options listed are 'Don't include in launch template' (selected), 'Create new key pair' (button), 'Specify a custom value...', 'Wordpress key' (disabled), and 'Don't include in launch template' (disabled). Below the dropdown, a note says 'When you specify a subnet, a network interface is automatically added to your template.' A 'Firewall (security groups)' section follows, with a note about security groups controlling traffic. At the bottom are 'Select existing security group' and 'Create security group' buttons, with 'Create security group' being highlighted. On the right, a summary panel shows the selected AMI ('Wordpress ami'), instance type ('t2.micro'), and storage ('1 volume(s) - 8 GiB'). A tooltip for the free tier is visible, and a large orange 'Create launch template' button is at the bottom right.

5. Choose a public subnet in network settings and create a security group allowing ssh and http ports.

The screenshot shows the AWS EC2 console with the same URL as the previous screenshot. The 'Network settings' section is now active. It shows a 'Subnet info' dropdown with 'subnet-0522518f37373a09a' selected, along with VPC, owner, availability zone, and CIDR information. Below this is a note about network interfaces. A 'Firewall (security groups)' section follows, with a note about security groups controlling traffic. At the bottom are 'Select existing security group' and 'Create security group' buttons, with 'Create security group' being highlighted. The 'Security group name - required' field contains 'SG for wordpress server'. Below it is a note about character restrictions. The 'Description - required' field contains 'Allows SSH access to developers'. Under 'VPC - required', the VPC is set to 'vpc-071ca095602023b83' with a CIDR of '172.31.0.0/16'. The 'Inbound security groups rules' section shows a single rule: 'Security group rule 1 (TCP, 80, 0.0.0.0/0)'. On the right, the summary panel remains the same, and a tooltip for the free tier is visible, with a large orange 'Create launch template' button at the bottom right.

The screenshot shows the 'Inbound security groups rules' section. It contains two rules:

- Security group rule 1 (TCP, 80, 0.0.0.0/0)**: Type: HTTP, Protocol: TCP, Port range: 80. Source type: Anywhere.
- Security group rule 2 (TCP, 22, 0.0.0.0/0)**: Type: ssh, Protocol: TCP, Port range: 22. Source type: Anywhere.

A warning message at the bottom left states: "⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only." A 'Create launch template' button is visible on the right.

6. Select the volume type as gp3 and select encrypted.

The screenshot shows the 'Storage (volumes)' section. It lists one EBS volume:

- Volume 1 (AMI Root) (Custom)**: Storage type: EBS, Device name: /dev/xvda, Snapshot: snap-0ad915bff381956a1.
- Size (GiB)**: 8, **Volume type**: gp3, **IOPS**: 3000.
- Delete on termination**: Yes, **Encrypted**: Not encrypted.
- Throughput**: 125.

A note at the bottom left says: "ⓘ Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage". A 'Create launch template' button is visible on the right.

7.Click on create launch template after reviewing every parameters.

The screenshot shows the 'Create Launch Template' wizard in the AWS EC2 console. The current step is 'EBS Volumes'. The configuration includes:

- Volume 1 (AMI Root) (Custom)**
- Storage type**: EBS
- Device name - required**: /dev/xvda
- Snapshot**: snap-0ad915bff381956a1
- Size (GiB)**: 8
- Volume type**: gp3
- IOPS**: 3000
- Delete on termination**: Yes
- Encrypted**: Encrypted
- KMS key**: Don't include in launch tem...
- Throughput**: 125

A note indicates that free-tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. A 'Create launch template' button is visible at the bottom right.

8.The launch template is successfully created.

The screenshot shows the 'Launch templates' page in the AWS EC2 console. A success message is displayed:

Success
Successfully created [Wordpress-Autoscaling \(lt-01554ce58fa4fca5b\)](#)

Next steps

- Launch an instance**
With On-Demand Instances, you pay for compute capacity by the second (for Linux, with a minimum of 60 seconds) or by the hour (for all other operating systems) with no long-term commitments or upfront payments. Launch an On-Demand Instance from your launch template.
- Create an Auto Scaling group from your template**
Amazon EC2 Auto Scaling helps you maintain application availability and allows you to scale your Amazon EC2 capacity up or down automatically according to conditions you define. You can use Auto Scaling to help ensure that you are running your desired number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lulls to reduce costs.
- Create Auto Scaling group**
- Create Spot Fleet**
A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance (of each instance type in each Availability Zone) is set by Amazon EC2, and adjusted gradually based on the long-term supply of and demand for Spot Instances. Spot instances are well-suited for data-analysis, batch jobs, background processing, and optional tasks.
- Create Spot Fleet**

A 'View launch templates' button is located at the bottom right.

9.Goto autoscaling group and click on create autoscaling group.

The screenshot shows the AWS EC2 Auto Scaling homepage. On the left, there's a navigation sidebar with various services like Dedicated Hosts, Scheduled Instances, Capacity Reservations, Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces), Load Balancing (Load Balancers, Target Groups), and Auto Scaling (Launch Configurations, Auto Scaling Groups). The 'Auto Scaling Groups' option is highlighted. The main content area features a large heading 'Amazon EC2 Auto Scaling helps maintain the availability of your applications'. Below it, a text block explains that Auto Scaling groups are collections of Amazon EC2 instances that enable automatic scaling and fleet management features. To the right, there's a 'Create Auto Scaling group' button. A diagram titled 'How it works' illustrates the scaling process with an 'Auto Scaling group' containing four instances, labeled with 'Minimum size', 'Desired capacity', and 'Maximum size', with arrows indicating 'Scale out as needed'. Other sections include 'Pricing' (with a note about no additional fees beyond service fees) and 'Getting started' (with links to 'What is Amazon EC2 Auto Scaling?' and 'Getting started with Amazon EC2 Auto Scaling').

10.Choose the launch template created and click next.

The screenshot shows the 'Choose launch template or configuration' step in the AWS Auto Scaling wizard. On the left, a sidebar lists steps: Step 1 (current), Step 2, Step 3 (optional), Step 4 (optional), Step 5 (optional), Step 6 (optional), and Step 7. The main area has a heading 'Choose launch template or configuration' with a 'Info' link. It instructs users to specify a launch template for all instances. Below this, there's a 'Name' section where 'Auto Scaling group name' is set to 'AS-Wordpress'. There's also a 'Launch template' section where 'Wordpress-Autoscaling' is selected from a dropdown. Other fields include 'Description' (empty), 'AMI ID' (empty), 'Launch template' (selected 'Wordpress-Autoscaling'), 'Security groups' (empty), 'Instance type' (set to 't2.micro'), and 'Request Spot Instances' (unchecked). A 'Switch to launch configuration' link is also present.

11. Choose default VPC and subnets and click next.

The screenshot shows the AWS EC2 Auto Scaling Group creation wizard at Step 4: Network configuration. On the left, a sidebar lists steps from 2 to 7. The main area is titled "Network Info" and contains a note about using multiple Availability Zones. A dropdown menu for "VPC" is open, showing "vpc-071ca095602023b83" and its subnet "172.31.0.0/16 Default". Below this is a "Create a VPC" button. A section for "Availability Zones and subnets" lists several subnets with checkboxes, all of which are checked. The subnets listed are:

- us-east-1a | subnet-0f432f40d150224c1 172.31.0.0/20 Default
- us-east-1b | subnet-0322518f37737a09a 172.31.80.0/20 Default
- us-east-1c | subnet-07ca7f7b3fbeb2f79 172.31.16.0/20 Default
- us-east-1d | subnet-074204f232be4ece6 172.31.32.0/20 Default
- us-east-1e | subnet-015973090fcc8ba74 172.31.48.0/20 Default
- us-east-1f | subnet-0c8396bc4713cc82b 172.31.64.0/20 Default

12. Choose 60 for health checks and click on next.

The screenshot shows the AWS EC2 Auto Scaling Group creation wizard at Step 5: Health checks. The sidebar shows steps 4 to 7. The main area has three radio button options: "No load balancer" (selected), "Attach to an existing load balancer", and "Attach to a new load balancer". Below this is a section titled "Health checks - optional". Under "Health check type", "EC2" is checked and "ELB" is unchecked. Under "Health check grace period", a input field shows "60 seconds". At the bottom, there's an "Additional settings - optional" section with "Monitoring" and "Default instance warmup" options, both of which have checkboxes that are unchecked. At the very bottom are "Cancel", "Previous", "Skip to review", and a large orange "Next" button.

13. Choose the capacities as 1 for all and click next.

The screenshot shows the AWS Auto Scaling Group creation wizard at Step 4. On the left sidebar, there are several optional steps: Step 3 (Configure advanced options), Step 4 (Configure group size and scaling policies), Step 5 (Add notifications), Step 6 (Add tags), and Step 7 (Review). The main panel is titled "Group size - optional". It contains fields for Desired capacity (set to 1), Minimum capacity (set to 1), and Maximum capacity (set to 1). Below this is a section titled "Scaling policies - optional" which includes a radio button for "Target tracking scaling policy" (disabled) and another for "None" (selected). At the bottom is an "Instance scale-in protection - optional" section with a checkbox for "Enable instance protection from scale in" (unchecked).

14. If you want to add notification select it and click next.

15. Review everything and click on 'create a autoscaling group'.

The screenshot shows the AWS Auto Scaling Group creation wizard at Step 7: Review. The main panel displays the configuration summary. It includes sections for "No scaling policy", "Instance scale-in protection" (with "Enable instance protection from scale in" unchecked), "Step 5: Add notifications" (with "Notifications" and "No notifications" sections), and "Step 6: Add tags" (with "Tags (0)" and a table for adding tags). At the bottom right is a prominent orange "Create Auto Scaling group" button.

16. Autoscaling group is successfully created.

The screenshot shows the AWS EC2 Auto Scaling Groups page. At the top, there are two informational messages: one about predictive scaling policy supporting custom metrics and another confirming the creation of 'AS-Wordpress'. Below this, the main table displays one Auto Scaling group:

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	A...
AS-Wordpress	Wordpress-Autoscaling Version Default	1	-	1	1	1	us...

At the bottom, it says "0 Auto Scaling groups selected".

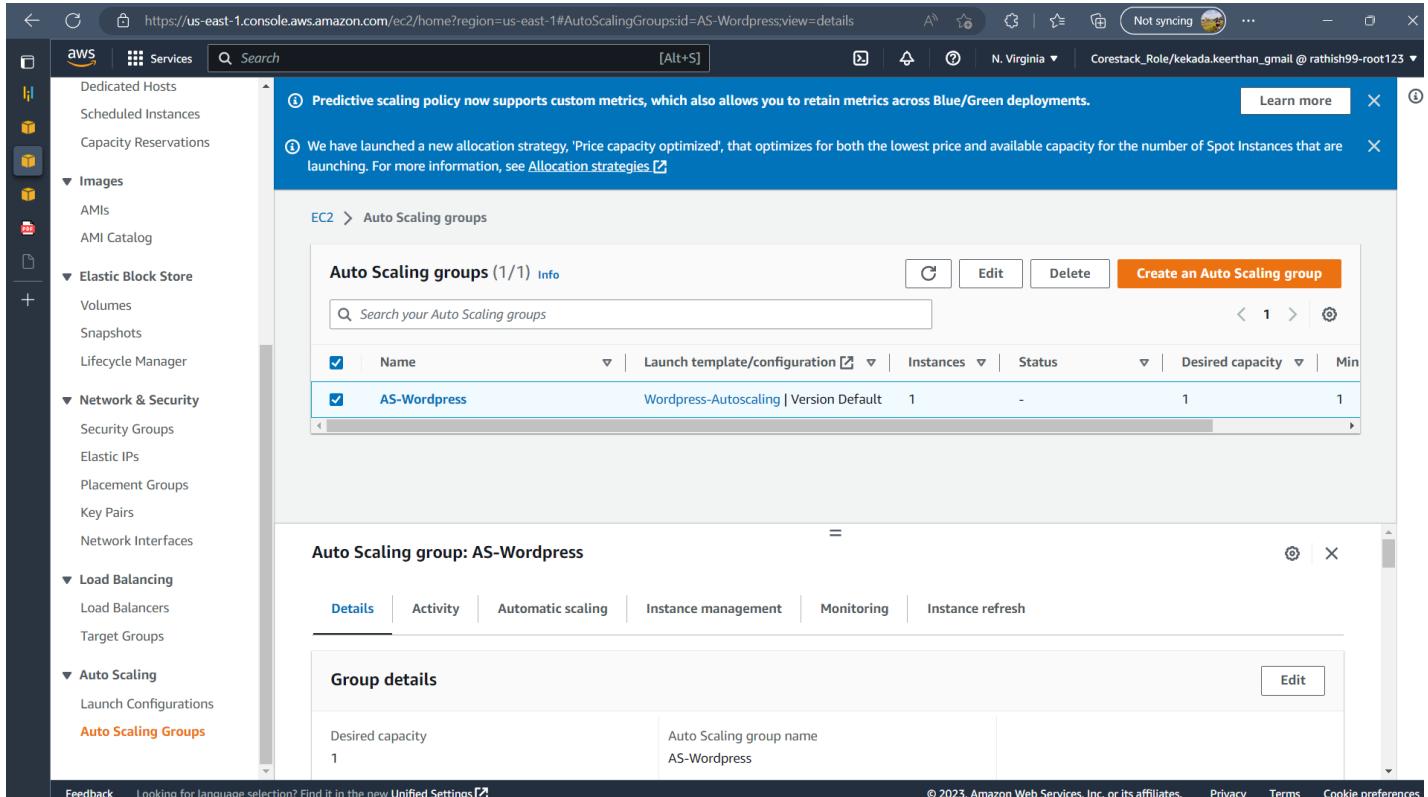
The screenshot shows the AWS EC2 Instances page. The left sidebar is expanded to show the 'Instances' section, which includes options like Instance Types, Launch Templates, and Capacity Reservations. The main table lists two instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
wordpress server	i-0b12d9067649e015a	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b
Wordpress server from AS	i-0d3ccdb1523791e07	Running	t2.micro	2/2 checks passed	No alarms	us-east-1f

Below the table, a modal window titled "Select an instance" is open, indicating the user is about to choose one of the listed instances.

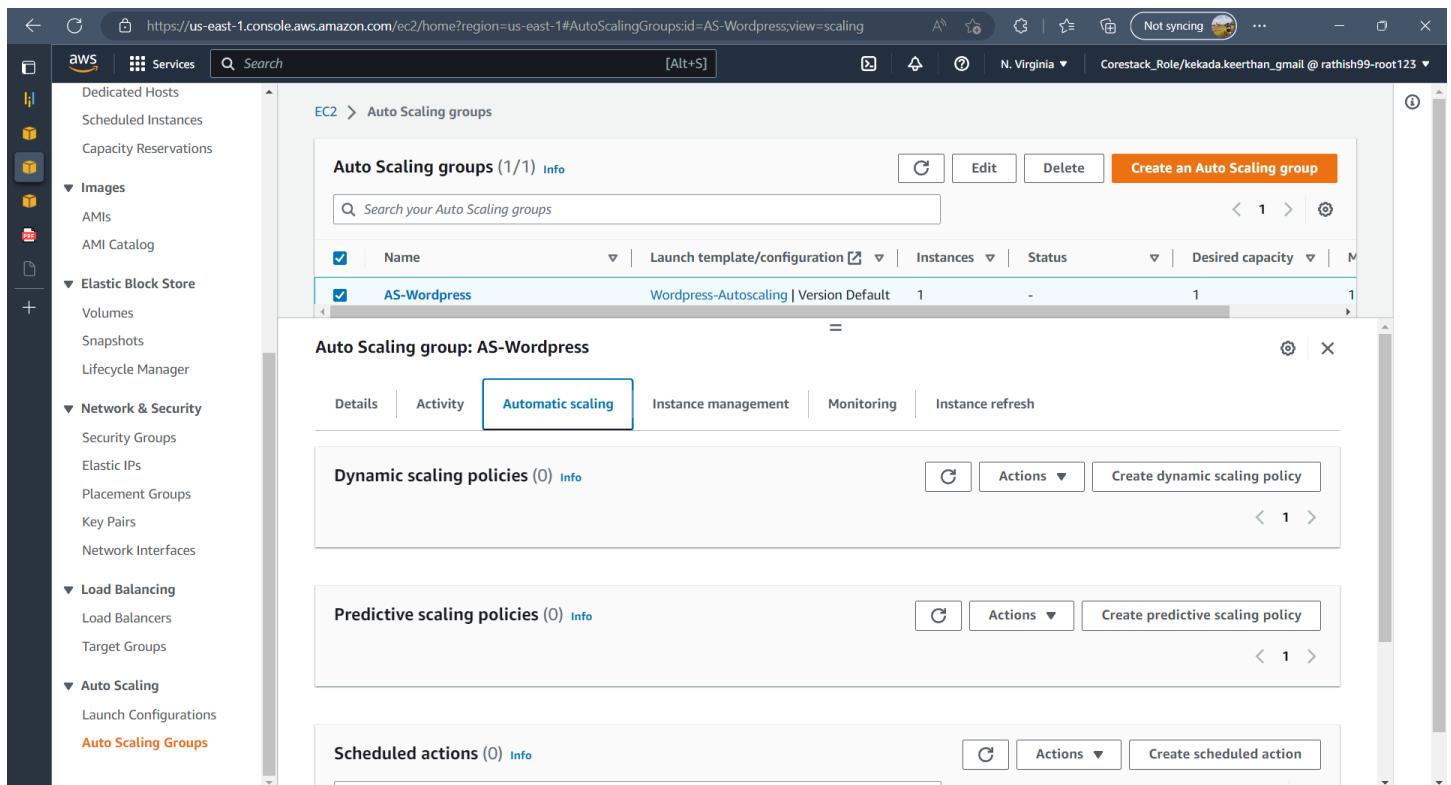
D}.Configure the new WordPress instance to shut down automatically

1.Goto autoscaling groups and select the instance created with autoscaling group.



The screenshot shows the AWS EC2 Auto Scaling Groups page. On the left, there's a navigation sidebar with various services like Dedicated Hosts, Scheduled Instances, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces, Load Balancing, Load Balancers, Target Groups, and Auto Scaling. Under Auto Scaling, 'Auto Scaling Groups' is selected. The main content area displays 'Auto Scaling groups (1/1) info'. A table lists one group: AS-Wordpress, which is associated with a Launch template/configuration named Wordpress-Autoscaling | Version Default. The status shows 1 instance at a desired capacity of 1. Below this, the 'Auto Scaling group: AS-Wordpress' details are shown, including the desired capacity (1) and the auto scaling group name (AS-Wordpress). There are tabs for Details, Activity, Automatic scaling, Instance management, Monitoring, and Instance refresh, with 'Automatic scaling' being the active tab.

2.Click on automatic scaling .



This screenshot is from the same EC2 Auto Scaling Groups page as the previous one, but the 'Automatic scaling' tab is now selected. The interface remains largely the same, showing the 'Auto Scaling groups (1/1) info' table with the single entry for AS-Wordpress. Below the table, the 'Auto Scaling group: AS-Wordpress' details are visible. The tabs for Details, Activity, Automatic scaling, Instance management, Monitoring, and Instance refresh are present, with 'Automatic scaling' being the active tab. Under the 'Automatic scaling' tab, there are sections for 'Dynamic scaling policies (0)', 'Predictive scaling policies (0)', and 'Scheduled actions (0)'. Each section has a 'Create [policy type] policy' button.

3. Goto scheduled actions and click on ‘create scheduled actions’.

The screenshot shows the AWS Management Console for the EC2 service, specifically the Auto Scaling Groups section. On the left sidebar, under the 'Auto Scaling' heading, 'Auto Scaling Groups' is selected. In the main content area, there is a table titled 'Auto Scaling groups (1/1)'. A single row is listed: 'AS-Wordpress' with a status of 'Wordpress-Autoscaling | Version Default'. Below this table, a section titled 'Auto Scaling group: AS-Wordpress' contains a 'Predictive scaling policies (0)' section. At the bottom of the page, there is a 'Scheduled actions (0)' section with a button labeled 'Create scheduled action'.

4. Give required details and click create.

The screenshot shows the 'Create scheduled action' dialog box. The 'Name' field is filled with 'Start'. A note below it says 'Provide at least one value for Desired, Min, or Max Capacity'. Under 'Desired capacity', the value '1' is selected. Under 'Min', the value '1' is selected. Under 'Max', the value '3' is selected. The 'Recurrence' dropdown is set to 'Every day' with the cron expression '(Cron) 0 9 * * *'. The 'Time zone' dropdown is set to 'Asia/Kolkata'. Below this, there is a 'Specific start time' section with fields for date ('2023/01/17'), time ('09:00'), and time zone ('Asia/Kolkata'). A 'Set End Time' button is present. At the bottom right of the dialog are 'Cancel' and 'Create' buttons.

The screenshot shows the AWS Auto Scaling Groups page. A modal window titled "Create scheduled action" is open. In the "Name" field, "stop" is entered. Below it, a note says "Provide at least one value for Desired, Min, or Max Capacity". Under "Desired capacity", the value "1" is set, with "Min" at 1 and "Max" at 3. The "Recurrence" dropdown is set to "Every day" with the cron expression "(Cron) 0 18 * * *". The "Time zone" is set to "Asia/Kolkata". A specific start time is configured for January 17, 2023, at 18:00. The "Create" button is visible at the bottom right of the modal.

5. Now the instance will start at 9:00 AM everyday and automatically shut down at 6:00 PM everyday.

The screenshot shows the AWS Auto Scaling Groups page. The "AS-Wordpress" group is selected. The "Scheduled actions" section lists two entries: "Start" and "stop". The "Start" action is set to run daily at 09:00 UTC, and the "stop" action is set to run daily at 18:00 UTC. Both actions have a "Desired capacity" of 1 and are configured for the "Asia/Kolkata" time zone.

Name	Start time	Recurrence	Time zone	Desired capacity
Start	2023 January 1...	0 9 * * *	Asia/Kolkata	1
stop	2023 January 1...	0 18 * * *	Asia/Kolkata	1

E}.Monitor the instance using Availability Monitoring feature of the R53

1.Goto route 53 and select hosted zone.

The screenshot shows the AWS Route 53 console. The left sidebar has a tree view with 'Route 53' selected. Under 'Hosted zones', it says '(0)' and there's a 'Create hosted zone' button. The main content area is titled 'Hosted zones (0)' and says 'No hosted zones'. It includes a search bar and a 'Create hosted zone' button.

2.Create a hosted zone using the domain name purchased ie.“keerthanappachu.in”.

The screenshot shows the AWS Route 53 console with a hosted zone named 'keerthanappachu.in'. The 'Records' tab is selected, showing two entries:

Record name	Type	Routing policy	Differences	Value/Route traffic to
keerthanapp...	NS	Simple	-	ns-1937.awsdns-50.co.uk. ns-621.awsdns-13.net. ns-409.awsdns-51.com. ns-1460.awsdns-54.org.
keerthanapp...	SOA	Simple	-	ns-1937.awsdns-50.co.uk. awsdns...

3.Edit and update the nameservers for the domain name ie.“keerthanappachu.in”.

keerthanappachu.in

Nameservers

Nameservers handle internet requests for your domain. You can use Hostinger nameservers or use custom nameservers to point to other hosting provider.

ns-1468.awsdns-55.org
ns-1749.awsdns-26.co.uk
ns-318.awsdns-39.com
ns-518.awsdns-00.net

Select Nameservers

Use Hostinger nameservers (recommended)
 Change nameservers

ns-1468.awsdns-55.org
ns-1749.awsdns-26.co.uk
ns-318.awsdns-39.com
ns-518.awsdns-00.net

Save Cancel

4.Now goto health checks and create health checks using details such as domain name and create health check.

5.The tab shows the website is healthy.

Health check with id 8db528d8-316a-4649-95db-030609fec8e5 has been created successfully

Name	Status	Description	Alarms	ID
Keerthan website	15 minutes ago now Healthy	http://keerthanappachu.in:80/	No alarms configured.	8db528d8-316a-4649-95db-030609fec8e5

Keerthan website

Latency