Project 1 - AWS SAA-C03

Scenario:

You have
been asked to set up a WordPress blog for your company per their defined
specifications.

During
business hours (9AM-6PM), the blogging team would like to use their own
development copy of the WordPress instance, so that any work they are doing
doesn’t impact the live copy.

You are required to create a plan
to achieve these goals using Amazon CloudFormation, creating a new AMI and Auto
Scaling.

**Key elements of
the project**:

Use Amazon CloudFormation to create an EC2 instance to run
WordPress with the following specification:

- Instance Type: T2.micro
- Create a new AMI of the WordPress instance.
- Configure Auto Scaling to launch a new WordPress instance
  during 9AM-6PM.

Solution:

To achieve the goal of setting up a WordPress blog with a development environment using Amazon CloudFormation, AMI creation, and Auto Scaling, here's a detailed plan:

First I create an instance for wordpress live copy:

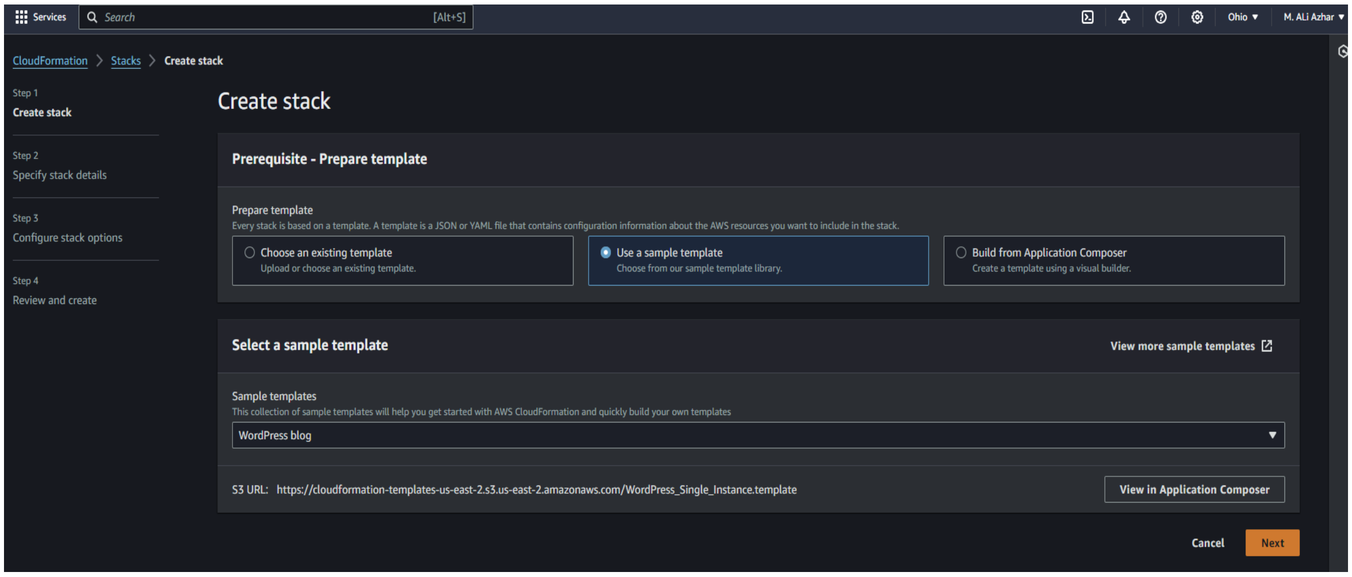
1. Select "CloudFormation" from the dashboard and then click on "Create New Stack".

2. Then I select use a sample template and from the dropdown of sample templates and choose "WordPress blog" and click Next.

3. I enter a name (**Project-01**) in the "Stack Name" box and complete the rest of the options like username and DB password etc and change the Instance Type to t2 micro.

4. In the KeyName section, I select **mytest-key** that I created before in **Ohio region**. Then click next and Review the settings and click "Create". Wait until the Stack process has a status of "COMPLETED".

Below are the screenshots of my AWS console.



Create stack

Prerequisite - Prepare template

Prepare template
Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

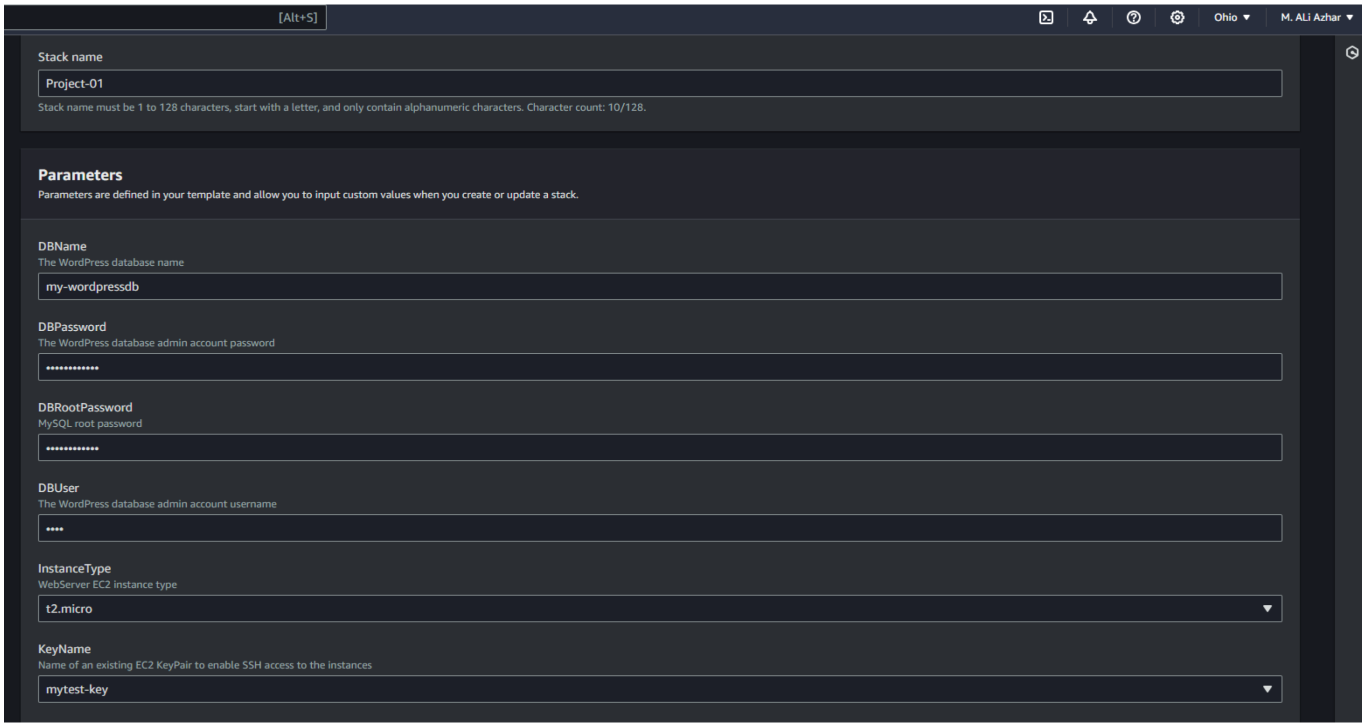
Choose an existing template — Upload or choose an existing template. [✓ Use a sample template — Choose from our sample template library.] Build from Application Composer — Create a template using a visual builder.

Select a sample template

Sample templates
This collection of sample templates will help you get started with AWS CloudFormation and quickly build your own templates
WordPress blog

S3 URL: https://cloudformation-templates-us-east-2.s3.us-east-2.amazonaws.com/WordPress_Single_Instance.template

Fig 1: Cloud formation Stack Creation



[Alt+S] Ohio M. ALi Azhar

Stack name

Project-01

Stack name must be 1 to 128 characters, start with a letter, and only contain alphanumeric characters. Character count: 10/128.

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
my-wordpressdb

DBPassword
The WordPress database admin account password
••••••••••••

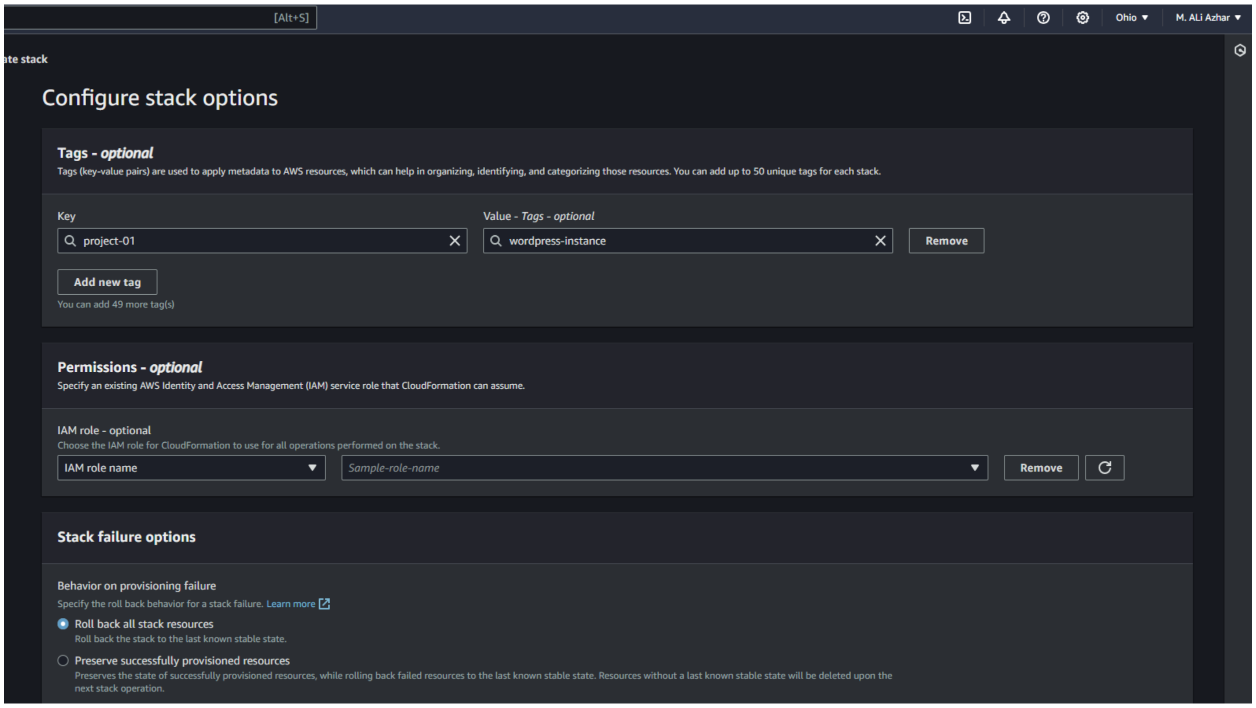
DBRootPassword
MySQL root password
••••••••••••

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
mytest-key

Fig 2: Cloud formation Stack Details



[Alt+S] Ohio M. ALi Azhar

ate stack

Configure stack options

Tags - *optional*

Tags (key-value pairs) are used to apply metadata to AWS resources, which can help in organizing, identifying, and categorizing those resources. You can add up to 50 unique tags for each stack.

Key: project-01

Value - *Tags - optional*: wordpress-instance

Remove

Add new tag

You can add 49 more tag(s)

Permissions - *optional*

Specify an existing AWS Identity and Access Management (IAM) service role that CloudFormation can assume.

IAM role - optional
Choose the IAM role for CloudFormation to use for all operations performed on the stack.
IAM role name | *Sample-role-name* | Remove

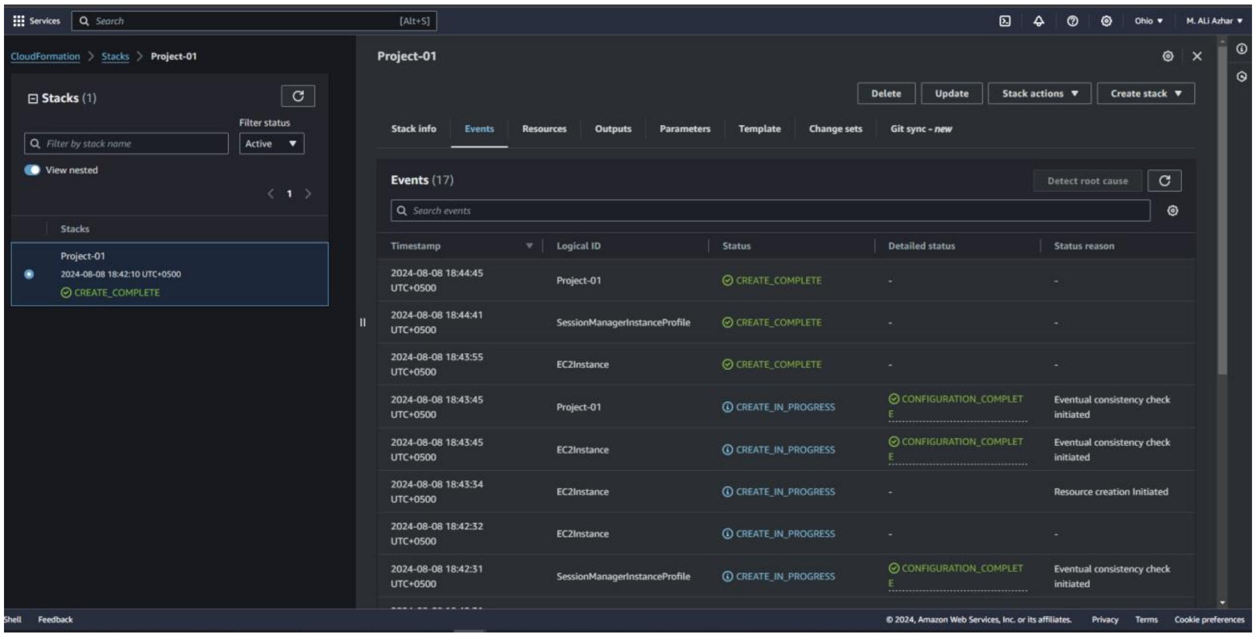
Stack failure options

Behavior on provisioning failure
Specify the roll back behavior for a stack failure. Learn more

● 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.

Fig 3: Stack tags



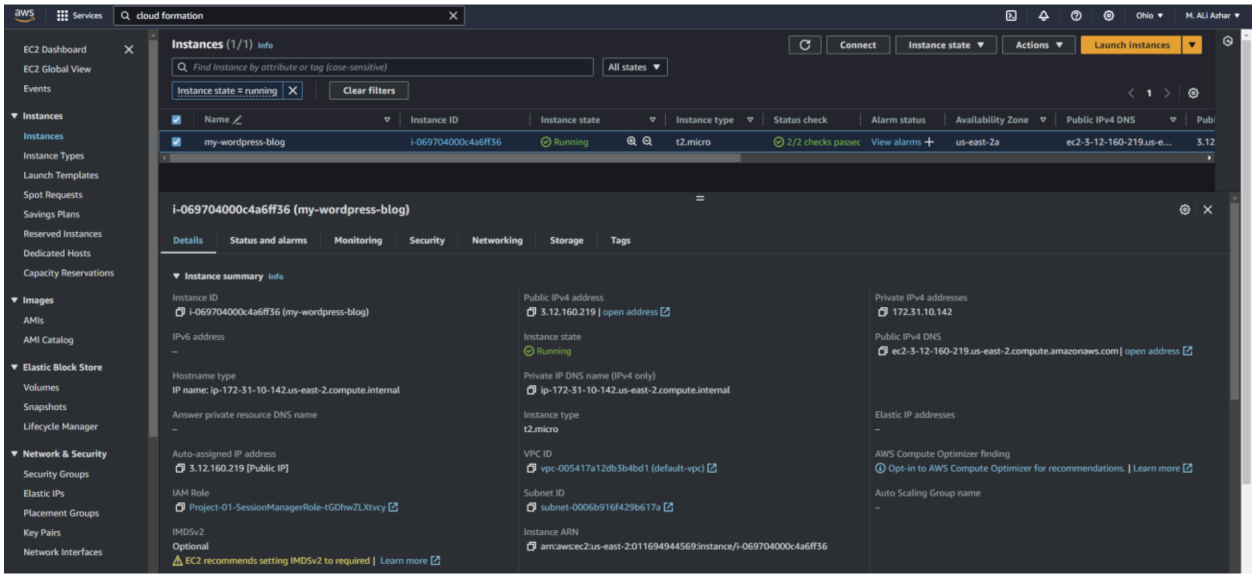
Project-01

Stacks (1) — Project-01, 2024-08-08 18:42:10 UTC+0500, CREATE_COMPLETE

Events (17)

Timestamp	Logical ID	Status	Detailed status	Status reason
2024-08-08 18:44:45 UTC+0500	Project-01	CREATE_COMPLETE	-	-
2024-08-08 18:44:41 UTC+0500	SessionManagerInstanceProfile	CREATE_COMPLETE	-	-
2024-08-08 18:43:55 UTC+0500	EC2Instance	CREATE_COMPLETE	-	-
2024-08-08 18:43:45 UTC+0500	Project-01	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE	Eventual consistency check initiated
2024-08-08 18:43:45 UTC+0500	EC2Instance	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE	Eventual consistency check initiated
2024-08-08 18:43:34 UTC+0500	EC2Instance	CREATE_IN_PROGRESS	-	Resource creation Initiated
2024-08-08 18:42:32 UTC+0500	EC2Instance	CREATE_IN_PROGRESS	-	-
2024-08-08 18:42:31 UTC+0500	SessionManagerInstanceProfile	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE	Eventual consistency check initiated

Fig 4: Project-01 stack status completed



Instances (1/1)

Instance state = running

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
my-wordpress-blog	i-069704000c4a6ff36	Running	t2.micro	2/2 checks passec	View alarms	us-east-2a	ec2-3-12-160-219.us-e...

i-069704000c4a6ff36 (my-wordpress-blog)

Instance summary

Instance ID: i-069704000c4a6ff36 (my-wordpress-blog)	Public IPv4 address: 3.12.160.219 | open address	Private IPv4 addresses: 172.31.10.142
IPv6 address: –	Instance state: Running	Public IPv4 DNS: ec2-3-12-160-219.us-east-2.compute.amazonaws.com | open address
Hostname type: IP name: ip-172-31-10-142.us-east-2.compute.internal	Private IP DNS name (IPv4 only): ip-172-31-10-142.us-east-2.compute.internal	
Answer private resource DNS name: –	Instance type: t2.micro	Elastic IP addresses: –
Auto-assigned IP address: 3.12.160.219 [Public IP]	VPC ID: vpc-005417a12db3b4bd1 (default-vpc)	AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role: Project-01-SessionManagerRole-tGDhwZLXtvcy	Subnet ID: subnet-0006b916f429b617a	Auto Scaling Group name: –
IMDSv2: Optional — EC2 recommends setting IMDSv2 to required | Learn more	Instance ARN: arn:aws:ec2:us-east-2:011694944569:instance/i-069704000c4a6ff36	

Fig 5: Wordpress Ec2 instance created from cloudformation Stack

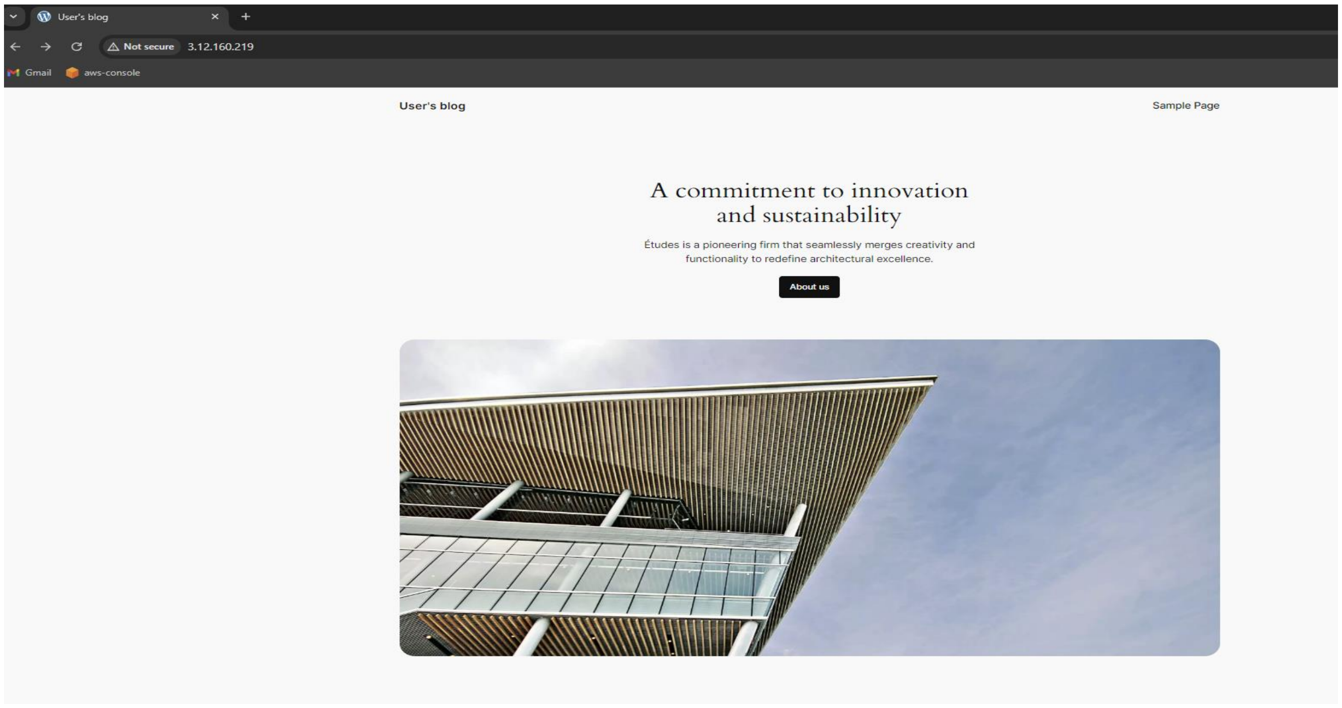
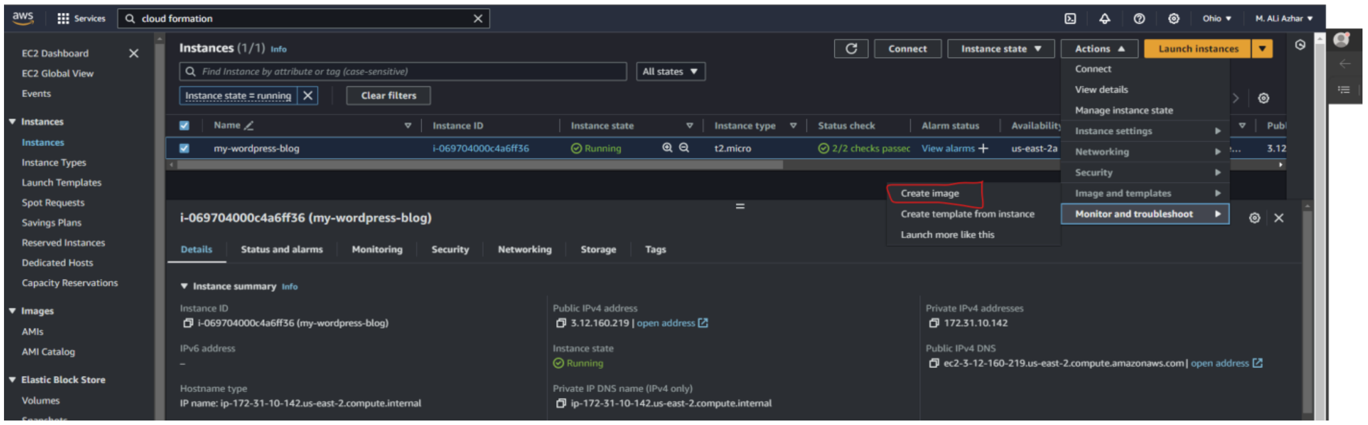

User's blog
Not secure 3.12.160.219
Gmail
aws-console
User's blog
Sample Page
A commitment to innovation and sustainability
Études is a pioneering firm that seamlessly merges creativity and functionality to redefine architectural excellence.
About us

Fig 6: Wordpress sample user blog page

AMI Creation:

- I select the newely created instance and click on Actions > Image > Create Image.
- Here I enter a name (**my-wordpress-blog**) for my AMI image and then click on Create Image.
- I switch to the AMI dashboard and wait until my new AMI has a status of available.



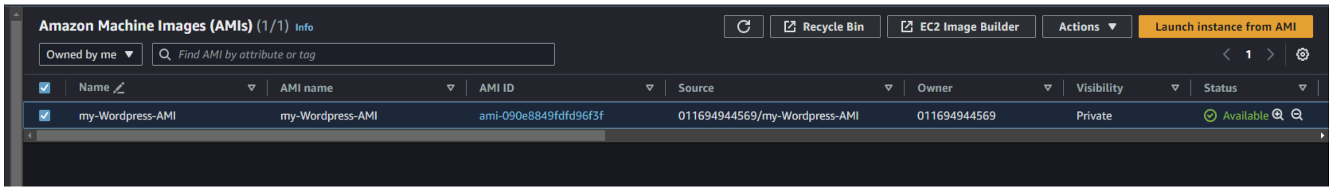
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability	Publ
my-wordpress-blog	i-069704000c4a6ff36	Running	t2.micro	2/2 checks passec	View alarms +	us-east-2a	3.12

i-069704000c4a6ff36 (my-wordpress-blog)

Instance summary Info

Instance ID: i-069704000c4a6ff36 (my-wordpress-blog)
Public IPv4 address: 3.12.160.219 | open address
Private IPv4 addresses: 172.31.10.142
IPv6 address: –
Instance state: Running
Public IPv4 DNS: ec2-3-12-160-219.us-east-2.compute.amazonaws.com | open address
Hostname type: IP name: ip-172-31-10-142.us-east-2.compute.internal
Private IP DNS name (IPv4 only): ip-172-31-10-142.us-east-2.compute.internal

Fig 7: Creation of AMI from my new instance



Name	AMI name	AMI ID	Source	Owner	Visibility	Status
my-Wordpress-AMI	my-Wordpress-AMI	ami-090e8849fdfd96f3f	011694944569/my-Wordpress-AMI	011694944569	Private	Available

Fig 8: Newly created Wordpress AMI status available

Configuration of Auto Scaling by using cloud formation:

First I create a cloud formation template with the below mentioned details in the yaml format.

1. **Define the Resources:**

- EC2 Instance: Use a T2.micro instance with our customized wordpress-blog AMI.
- Security Group: Configure the security group to allow HTTP (port 80) and SSH (port 22) access.
- Auto Scaling Group: Configure the Auto Scaling Group for the development environment.
- Launch Configuration: Create a launch configuration that uses my custom AMI.

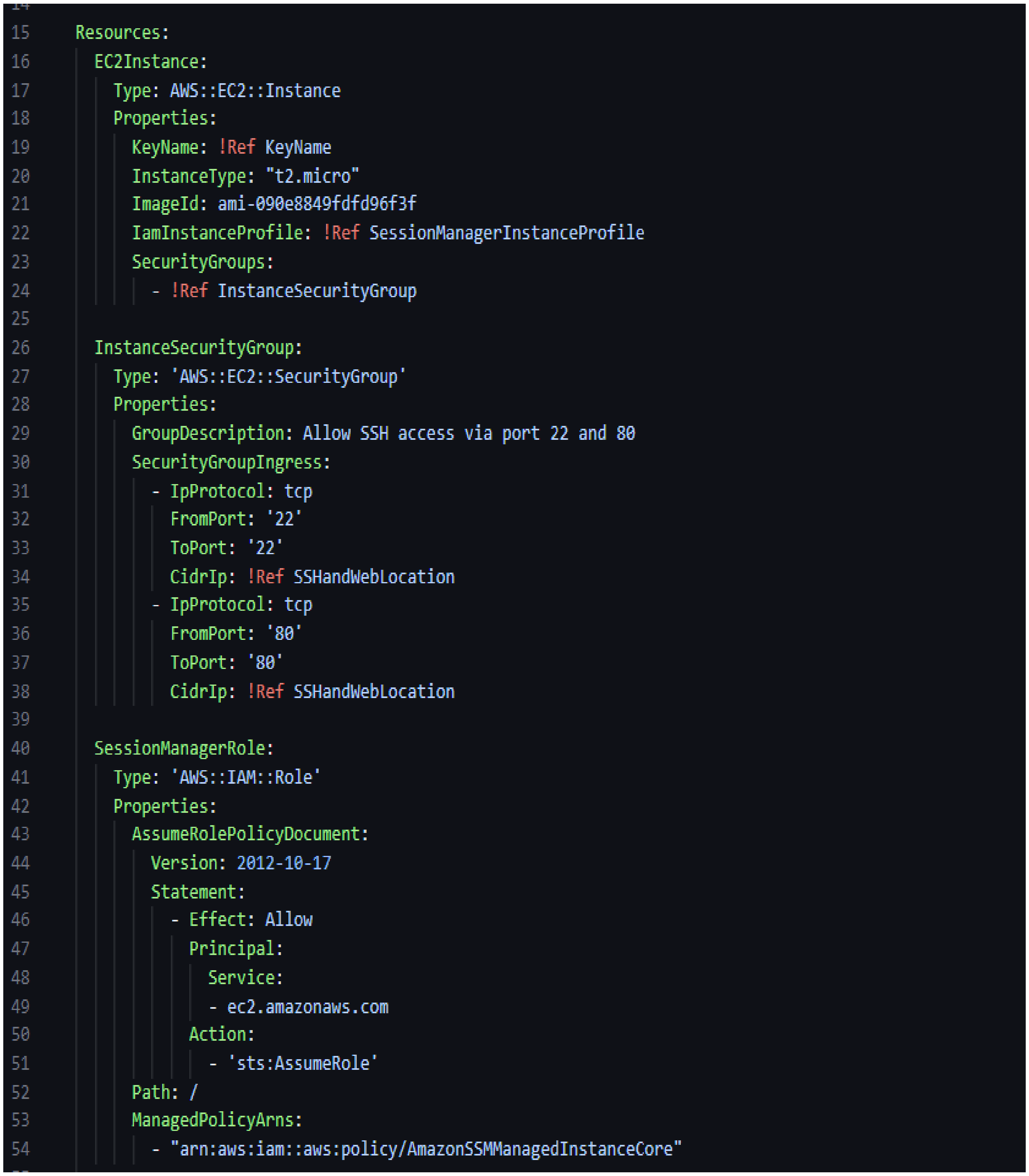
2. **CloudFormation Template Structure:**

- **Parameters:** Define parameters for instance type, key pair name, and AMI ID.
- **Resources:** Define the EC2 instance, security group, and Auto Scaling Group.
- **Outputs:** Provide the public IP address of the WordPress instance.

3. **Auto Scaling Group:**

- I define an Auto Scaling Group that launches a new instance during business hours (9AM-6PM).
- Use a scheduled action in Auto Scaling to start and stop the development instance and set the desired capacity 2, Minimum capacity 1 and Maximum capacity 2.

After successfully completed and tested, I terminated all the instances and delete all resources to avoid any charges from AWS.

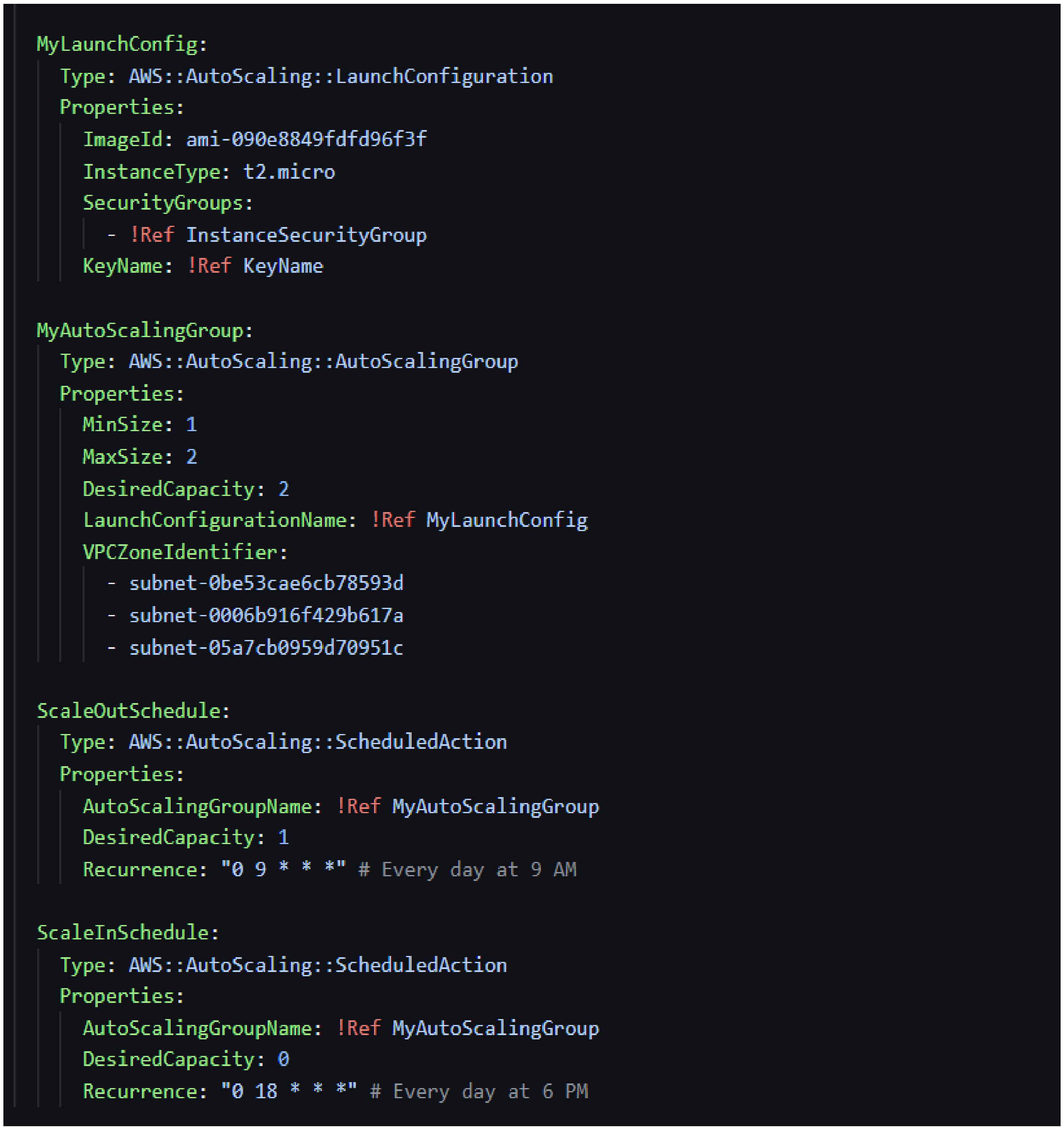


```yaml
Resources:
  EC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      KeyName: !Ref KeyName
      InstanceType: "t2.micro"
      ImageId: ami-090e8849fdfd96f3f
      IamInstanceProfile: !Ref SessionManagerInstanceProfile
      SecurityGroups:
        - !Ref InstanceSecurityGroup

  InstanceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Allow SSH access via port 22 and 80
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: '22'
          ToPort: '22'
          CidrIp: !Ref SSHandWebLocation
        - IpProtocol: tcp
          FromPort: '80'
          ToPort: '80'
          CidrIp: !Ref SSHandWebLocation

  SessionManagerRole:
    Type: 'AWS::IAM::Role'
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Principal:
              Service:
              - ec2.amazonaws.com
            Action:
              - 'sts:AssumeRole'
      Path: /
      ManagedPolicyArns:
        - "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"
```

Fig 9: Cloud Formation Template for Auto-Scalling



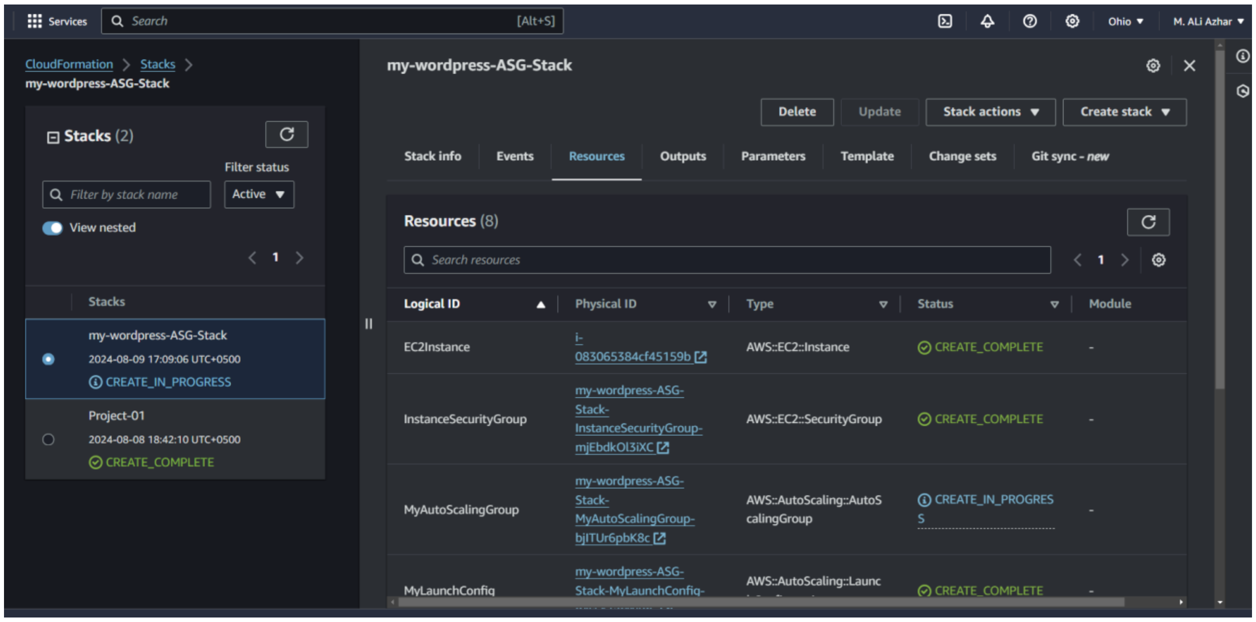
```
MyLaunchConfig:
  Type: AWS::AutoScaling::LaunchConfiguration
  Properties:
    ImageId: ami-090e8849fdfd96f3f
    InstanceType: t2.micro
    SecurityGroups:
      - !Ref InstanceSecurityGroup
    KeyName: !Ref KeyName

MyAutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    MinSize: 1
    MaxSize: 2
    DesiredCapacity: 2
    LaunchConfigurationName: !Ref MyLaunchConfig
    VPCZoneIdentifier:
      - subnet-0be53cae6cb78593d
      - subnet-0006b916f429b617a
      - subnet-05a7cb0959d70951c

ScaleOutSchedule:
  Type: AWS::AutoScaling::ScheduledAction
  Properties:
    AutoScalingGroupName: !Ref MyAutoScalingGroup
    DesiredCapacity: 1
    Recurrence: "0 9 * * *" # Every day at 9 AM

ScaleInSchedule:
  Type: AWS::AutoScaling::ScheduledAction
  Properties:
    AutoScalingGroupName: !Ref MyAutoScalingGroup
    DesiredCapacity: 0
    Recurrence: "0 18 * * *" # Every day at 6 PM
```

Fig 10: Cloud Formation Template to schedule AutoScaling



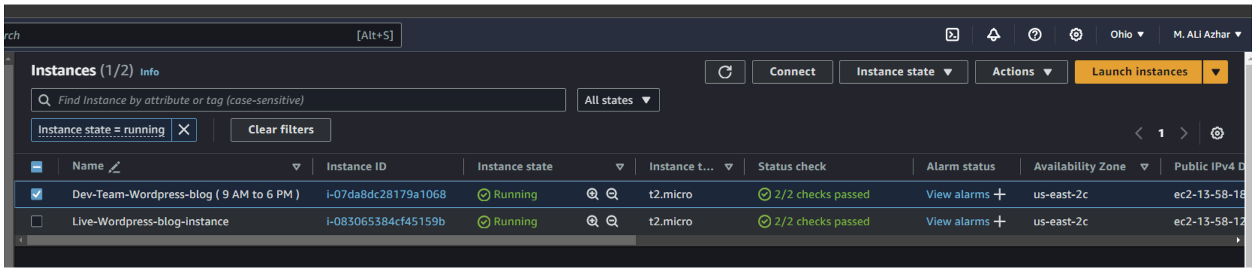
Stacks

Stack	Created	Status
my-wordpress-ASG-Stack	2024-08-09 17:09:06 UTC+0500	CREATE_IN_PROGRESS
Project-01	2024-08-08 18:42:10 UTC+0500	CREATE_COMPLETE

my-wordpress-ASG-Stack

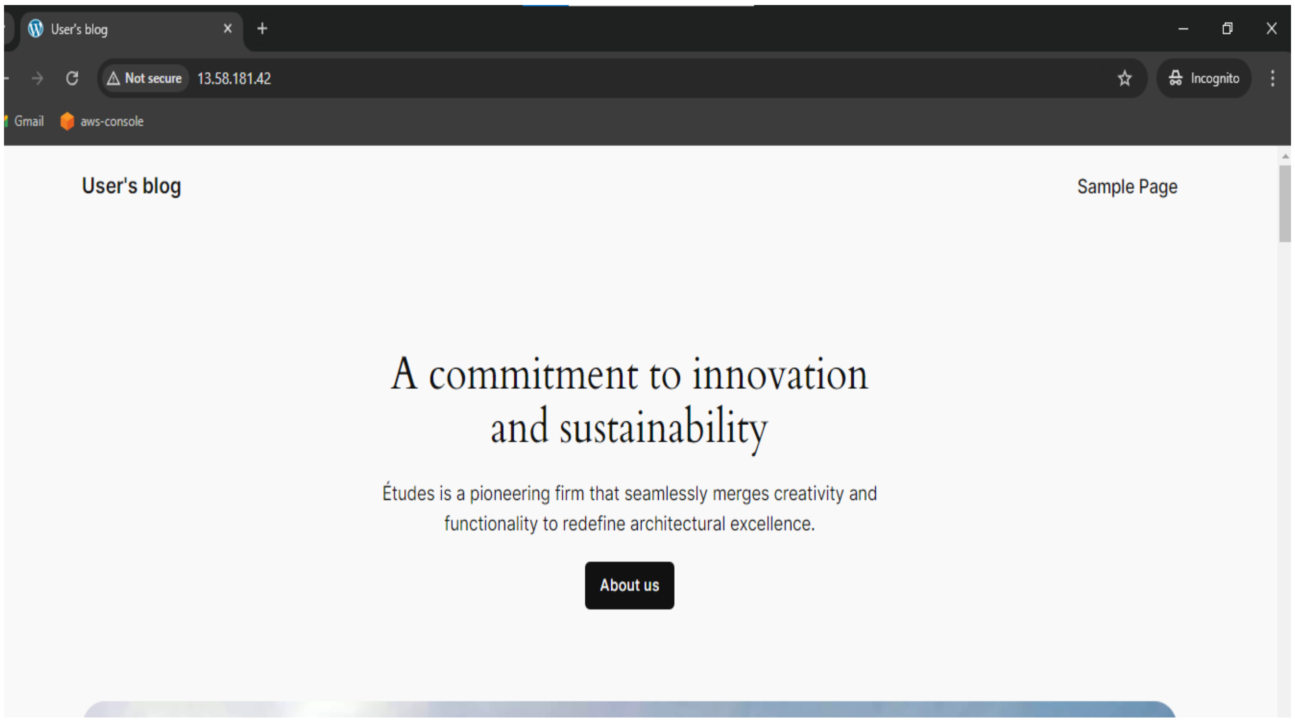
Logical ID	Physical ID	Type	Status	Module
EC2Instance	i-083065384cf45159b	AWS::EC2::Instance	CREATE_COMPLETE	-
InstanceSecurityGroup	my-wordpress-ASG-Stack-InstanceSecurityGroup-mjEbdkOl3iXC	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
MyAutoScalingGroup	my-wordpress-ASG-Stack-MyAutoScalingGroup-bjITUr6pbK8c	AWS::AutoScaling::AutoScalingGroup	CREATE_IN_PROGRESS	-
MyLaunchConfig	my-wordpress-ASG-Stack-MyLaunchConfig-	AWS::AutoScaling::Launc	CREATE_COMPLETE	-

Fig 11: Autoscalling cloud formation stack creation



	Name	Instance ID	Instance state	Instance t...	Status check	Alarm status	Availability Zone	Public IPv4 D
☑	Dev-Team-Wordpress-blog (9 AM to 6 PM)	i-07da8dc28179a1068	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2c	ec2-13-58-18
☐	Live-Wordpress-blog-instance	i-083065384cf45159b	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2c	ec2-13-58-12

Fig 12: Dev instance created successfully at 9AM



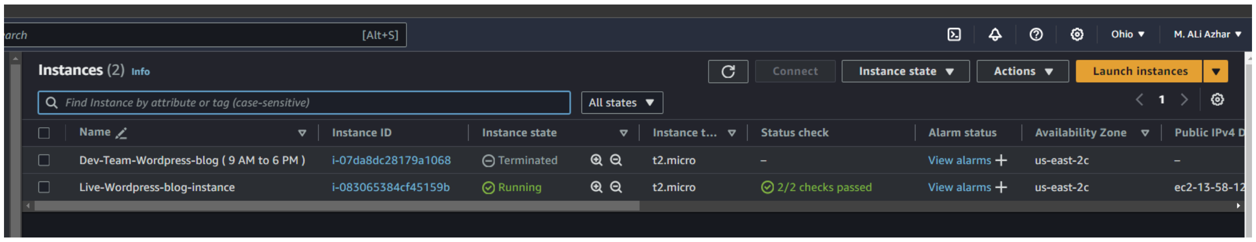
User's blog   Sample Page

A commitment to innovation and sustainability

Études is a pioneering firm that seamlessly merges creativity and functionality to redefine architectural excellence.

About us

Fig 13: Dev instance sample page



	Name	Instance ID	Instance state	Instance t...	Status check	Alarm status	Availability Zone	Public IPv4 D
☐	Dev-Team-Wordpress-blog (9 AM to 6 PM)	i-07da8dc28179a1068	Terminated	t2.micro	–	View alarms +	us-east-2c	–
☐	Live-Wordpress-blog-instance	i-083065384cf45159b	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2c	ec2-13-58-12

Fig 14: Dev instance terminated at 6PM