



Hands-on tutorials

Migrate an ASP.NET Web Application to Elastic Beanstalk



Migrate an ASP.NET Web Application to Elastic Beanstalk: Hands-on tutorials

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Migrate an ASP.NET Web Application to Elastic Beanstalk	i
Overview	1
What you'll accomplish	1
Prerequisites	2
Implementation	2
Conclusion	22

Migrate an ASP.NET Web Application to Elastic Beanstalk

AWS experience	Intermediate
Time to complete	15 minutes
Cost to complete	Running this tutorial for an hour with two t3.medium instances in the US East (N. Virginia) Region (us-east-1) will cost \$0.12 total (on demand). One day will cost \$2.88; one month will cost about \$84.40.
Requires	AWS account with IAM permissions to create an EC2 instance, key pair, security group, IAM user, and an Elastic Beanstalk environment.  Note Accounts created within the past 24 hours might not yet have access to the services required for this tutorial.
Last updated	March 30, 2023

Overview

The purpose of this tutorial is to migrate a sample ASP.NET web application to a fully managed [AWS Elastic Beanstalk](#) environment using the interactive Windows Web Application Migration Assistant (WWAMA). Additional information about the Windows Web Application Migration Assistant is available [here](#).

What you'll accomplish

- In this tutorial, you will migrate a sample ASP.NET web application to a fully managed AWS Elastic Beanstalk environment.

Prerequisites

You will need an AWS account and [IAM](#) permissions to create an [EC2](#) instance, key pair, security group, IAM user, and an Elastic Beanstalk environment. This tutorial will deploy an AWS CloudFormation template that automatically provisions the sample website on an EC2 instance that will be the source web application for the migration.

Implementation

Step 1: Sign up for AWS

- The CloudFormation template used in this tutorial launches a t3.medium EC2 instance, which is **not** included in the Free Tier. You can estimate EC2 costs on the [EC2 pricing page](#).

Already have an account? [Sign in](#).

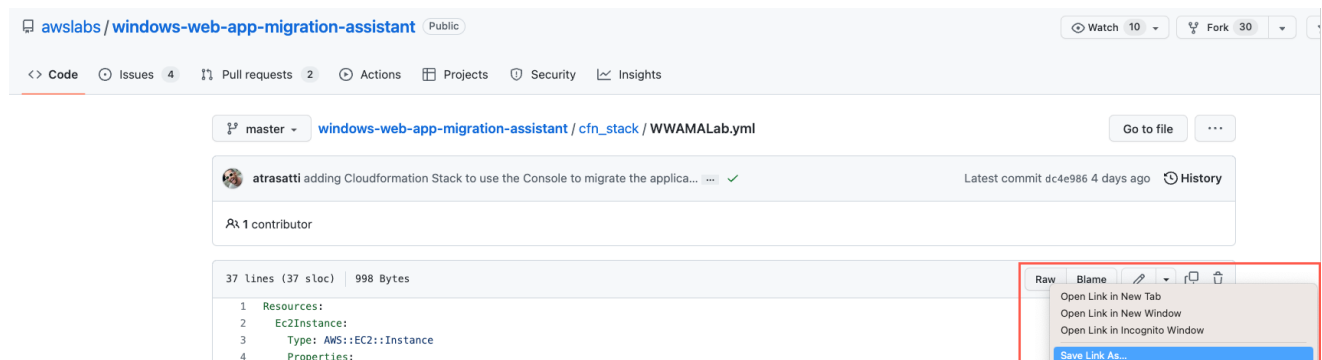
Step 2: Set up and configure the stack

- Use CloudFormation to launch the EC2 instance that will host the sample website. Then, set up the required IAM permissions.

a. Launch an EC2 instance

Use CloudFormation to launch an EC2 instance in the US-East-1 Region.

First, you will need to download the [YAML file](#) we will use for the CloudFormation template from GitHub. In GitHub, open the context (right-click) menu of the **Raw** button at the top of the file, select **Save Link As**. Choose the location on your computer where you want to save the file, and select **Save**.



b. Create the stack

Once you have saved the file locally, open the [Create stack](#) page within the AWS CloudFormation console. In the **Specify template** section, select **Upload a template file**.

Choose the **Choose file** button and navigate to the YAML file on your local machine and select **Open**.

Then choose **Next**.

CloudFormation > Stacks > Create stack

Step 1
Create stack

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review

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.

☒ Template is ready ☐ Use a sample template ☐ Create template in Designer

Specify template

A template is a JSON or YAML file that describes your stack's resources and properties.

Template source
Selecting a template generates an Amazon S3 URL where it will be stored.

☐ Amazon S3 URL ☒ Upload a template file

Upload a template file

WWAMALab.yml
JSON or YAML formatted file

S3 URL: <https://s3.us-east-1.amazonaws.com/ct-templates-> WWAMALab.yml

View in Designer

Cancel **Next**

c. Add a key pair

Select an existing key pair or [create a key pair](#) if you do not have one.

Then choose **Next**.

CloudFormation > Stacks > Create stack

Step 1
Create stack

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review WWAMASStack

Specify stack details

Stack name

Stack name

WWAMASStack

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.

KeyPair
Key Pair for this Instance

Select AWS::EC2::KeyPair::KeyName

Cancel Previous **Next**

d. Review the configuration

On the **Configure stack** options screen, choose **Next**. At the bottom of the **Review** screen, choose **Submit**.

CloudFormation > Stacks > Create stack

Step 1
Create stack

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review WWAMASStack

Review WWAMASStack

Step 1: Specify template Edit

Template

Template URL
https://public-read-location.s3.amazonaws.com/WWAMALab.yml

Stack description
-

Step 2: Specify stack details Edit

Parameters

Q Search parameters < 1 >

Stack creation options

Timeout
-

Termination protection
Disabled

► Quick-create link

Create change set Cancel Previous Submit

e. Verify completion

Once the stack has been created, you will see its status change to **CREATE_COMPLETE**.

WWAMASStack

Delete Update Stack actions ▼ Create stack ▼

Stack info **Events** Resources Outputs Parameters Template Change sets

Events (8) Refresh

Q Search events Settings

Timestamp ▼	Logical ID	Status	Status reason
2022-11-23 11:41:10 UTC-0500	WWAMASStack	🟢 CREATE_COMPLETE	-

Step 3: Create the IAM user

1. Log into the console and add a user

Log in to the [IAM console](#).

In the left navigation pane, select **Users**, then choose **Add users**.

Enter the **User name** **MigrationUser**, check the box for **Programmatic access**, then choose **Next:Permissions**.

Add user



Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name*

[+ Add another user](#)

Select AWS access type

Select how these users will primarily access AWS. If you choose only programmatic access, it does NOT prevent users from accessing the console using an assumed role. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

- Select AWS credential type*
- ☒ **Access key - Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.
 - ☐ **Password - AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

2. Attach Elastic Beanstalk policy

Select **Attach existing policies directly**, and enter **beanstalk** in the search bar to filter the policies. Select the **checkbox** next to **AdministratorAccess-AWSElasticBeanstalk**.

Add user

1

2

3

4

5

▼ Set permissions



Add user to group

Copy permissions from
existing userAttach existing policies
directly

Create policy



Filter policies ▼

Q beanstalk

Showing 14 results

	Policy name ▼	Type	Used as
☒	 AdministratorAccess-AWSElasticBeanstalk	AWS managed	None
☐	 AWSElasticBeanstalkCustomPlatformforEC2Role	AWS managed	None
☐	 AWSElasticBeanstalkEnhancedHealth	AWS managed	Permissions policy (1)
☐	 AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy	AWS managed	None
☐	 AWSElasticBeanstalkMulticontainerDocker	AWS managed	Permissions policy (1)
☐	 AWSElasticBeanstalkReadOnly	AWS managed	None
☐	 AWSElasticBeanstalkRoleCore	AWS managed	None

3. Attach IAM policy

Enter **iam** in the search bar and select the **checkbox** next to **IAMReadOnlyAccess**.

Add user

1

2

3

4

5

▼ Set permissions



Add user to group



Copy permissions from existing user



Attach existing policies directly

Create policy



Filter policies ▼

Q iam

Showing 11 results

	Policy name ▼	Type	Used as
☐	▶ AWSLoadBalancerControllerIAMPolicy	Customer managed	None
☐	▶ AWSQuickSightIAMPolicy	Customer managed	Permissions policy (1)
☐	▶ AWSQuickSightListIAM	AWS managed	None
☐	▶ IAMAccessAdvisorReadOnly	AWS managed	None
☐	▶ IAMAccessAnalyzerFullAccess	AWS managed	None
☐	▶ IAMAccessAnalyzerReadOnlyAccess	AWS managed	None
☐	▶ IAMFullAccess	AWS managed	None
☒	▶ IAMReadOnlyAccess	AWS managed	None
☐	▶ IAMSelfManageServiceSpecificCredentials	AWS managed	None

4. Attach S3 policy

Enter **s3** in the search bar and select the **checkbox** next to **AmazonS3FullAccess**.

Add user

1

2

3

4

5

▼ Set permissions



Add user to group



Copy permissions from existing user



Attach existing policies directly

Create policy



Filter policies ▼

Q s3

Showing 13 results

	Policy name ▼	Type	Used as
☐	▶ AmazonDMSRedshiftS3Role	AWS managed	None
☒	▶ AmazonS3FullAccess	AWS managed	Permissions policy (9)
☐	▶ AmazonS3ObjectLambdaExecutionRolePolicy	AWS managed	None
☐	▶ AmazonS3OutpostsFullAccess	AWS managed	None

5. Review configuration and create user

Choose **Next:Tags**, then **Next:Review**, and then **Create user**.

Add user



Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name	MigrationIser
AWS access type	Programmatic access - with an access key
Permissions boundary	Permissions boundary is not set

Permissions summary

The following policies will be attached to the user shown above.

Type	Name
Managed policy	AdministratorAccess-AWSElasticBeanstalk
Managed policy	IAMUserChangePassword
Managed policy	AmazonS3FullAccess

Tags

No tags were added.

[Cancel](#)[Previous](#)[Create user](#)

6. Download credentials

After the user is created, choose the **Download .csv** button when the prompt appears.

Add user

1

2

3

4

5



Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: [https://\[redacted\].signin.aws.amazon.com/console](https://[redacted].signin.aws.amazon.com/console)



Download .csv

	User	Access key ID	Secret access key
▶	✓ MigrationUser	[redacted]	***** Show

Step 4: Log in to the EC2 console and set up to run the WWAMA tool

1. Log into the EC2 console

Log in to [EC2 console](#).

Select the WWAMA instance and choose **Connect**.

The screenshot shows the AWS Management Console's EC2 Instances page. At the top, there are buttons for 'Refresh', 'Connect', 'Instance state', and 'Actions'. Below these is a search bar with the placeholder text 'Find instance by attribute or tag (case-sensitive)'. A filter is applied: 'Instance state = running'. A table lists the instances:

☒	Name	Instance ID	Instance state	Instance type
☒	WWAMA Lab	i-038b649ad2f4ce28b	Running	t2.micro

2. Connect to the instance

Select the **RDP client** tab, then choose the **Download remote desktop file** button and save the RDP file. Then choose **Get password**.

Connect to instance [Info](#)

Connect to your instance i-038b649ad2f4ce28b (WWAMA Lab) using any of these options

Session Manager

RDP client

EC2 serial console

Instance ID

 i-038b649ad2f4ce28b (WWAMA Lab)

Connection Type

☒ **Connect using RDP client**
Download a file to use with your RDP client and retrieve your password.

☐ **Connect using Fleet Manager**
Connect to your instance using Fleet Manager Remote Desktop.

You can connect to your Windows instance using a remote desktop client of your choice, and by downloading and running the RDP shortcut file below:

 **Download remote desktop file**

When prompted, connect to your instance using the following details:

Public DNS	User name
 ec2-44-226-226-22.compute-1.amazonaws.com	 Administrator

Password [Get password](#)

 If you've joined your instance to a directory, you can use your directory credentials to connect to your instance.

3. Upload private key

Choose **Upload private key file** to upload your private key, then choose **Decrypt password** to get your Windows Server password. You will see the password in plain text.

Copy it because you will need it in the next step.

Get Windows password [Info](#)

Use your private key to retrieve and decrypt the initial Windows administrator password for this instance.

Instance ID

 **i-038b649ad2f4ce28b** (WWAMA Lab)

Key pair associated with this instance

 **ssh-key-2019-08-22**

Private key

Either upload your private key file or copy and paste its contents into the field below.

 **Upload private key file**

 **ssh-key-2019-08-22**.pem
1.692KB

Private key contents - *optional*

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA...
-----END RSA PRIVATE KEY-----
```

Cancel

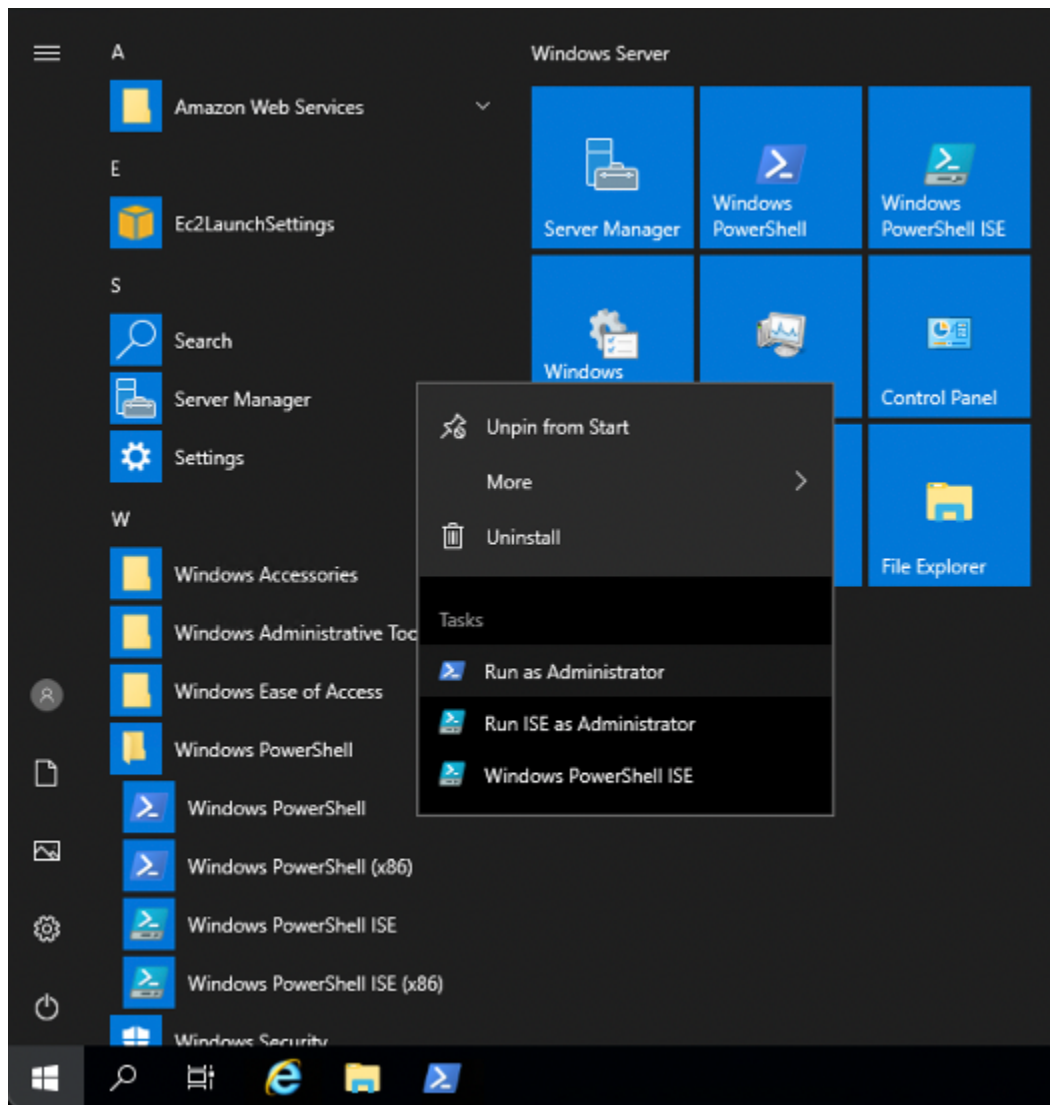
Decrypt password

- #### 4. Log into the instance

Log in to the EC2 instance using the RDP file you saved earlier and provide your password.

- ## 5. Open a PowerShell terminal

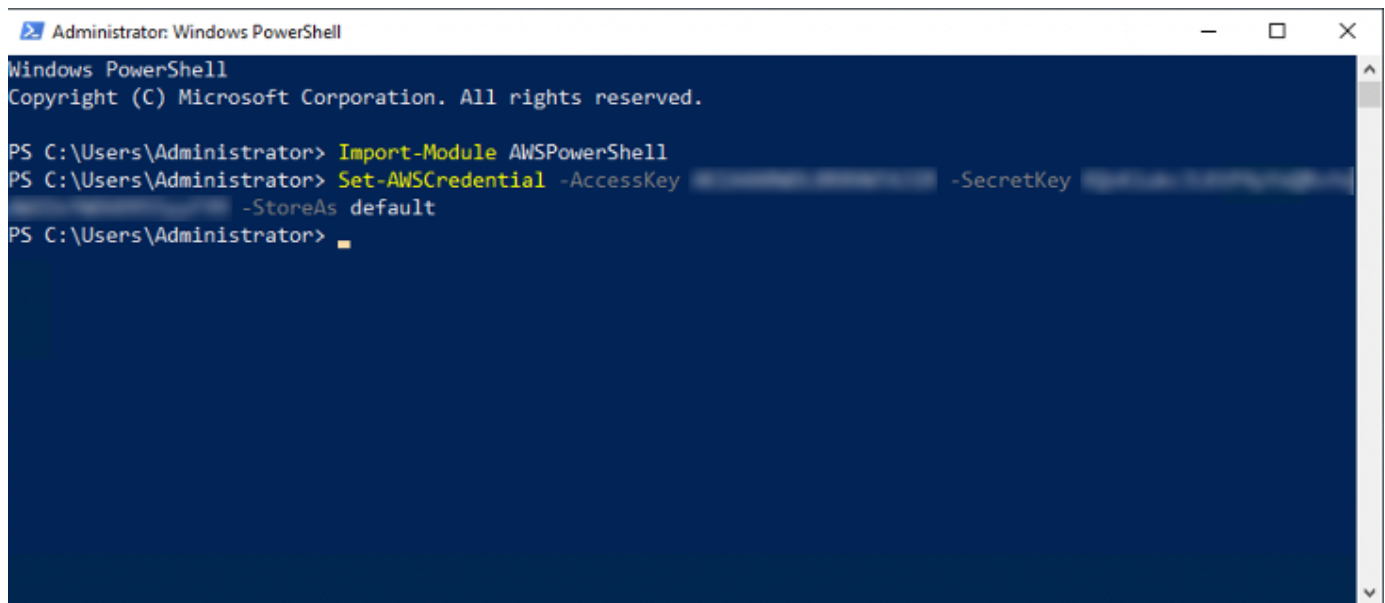
Open a PowerShell terminal as an Administrator by choosing the Windows icon in the lower left corner of the screen, open the **Windows PowerShell** folder, open the context (right-click) menu for **Windows PowerShell**, and then choose **Run as Administrator**.



6. Configure credentials

In the PowerShell window, run the commands provided in the PowerShell sample to configure the AWS credentials. Replace **ACCESS_KEY** and **SECRET_ACCESS_KEY** with the values in the CSV file you downloaded earlier during the creation of the **MigrationUser**.

```
PS C:\> Import-Module AWSPowerShell
PS C:\> Set-AWSCredential -AccessKey ACCESS_KEY -SecretKey SECRET_ACCESS_KEY -
StoreAs default
```



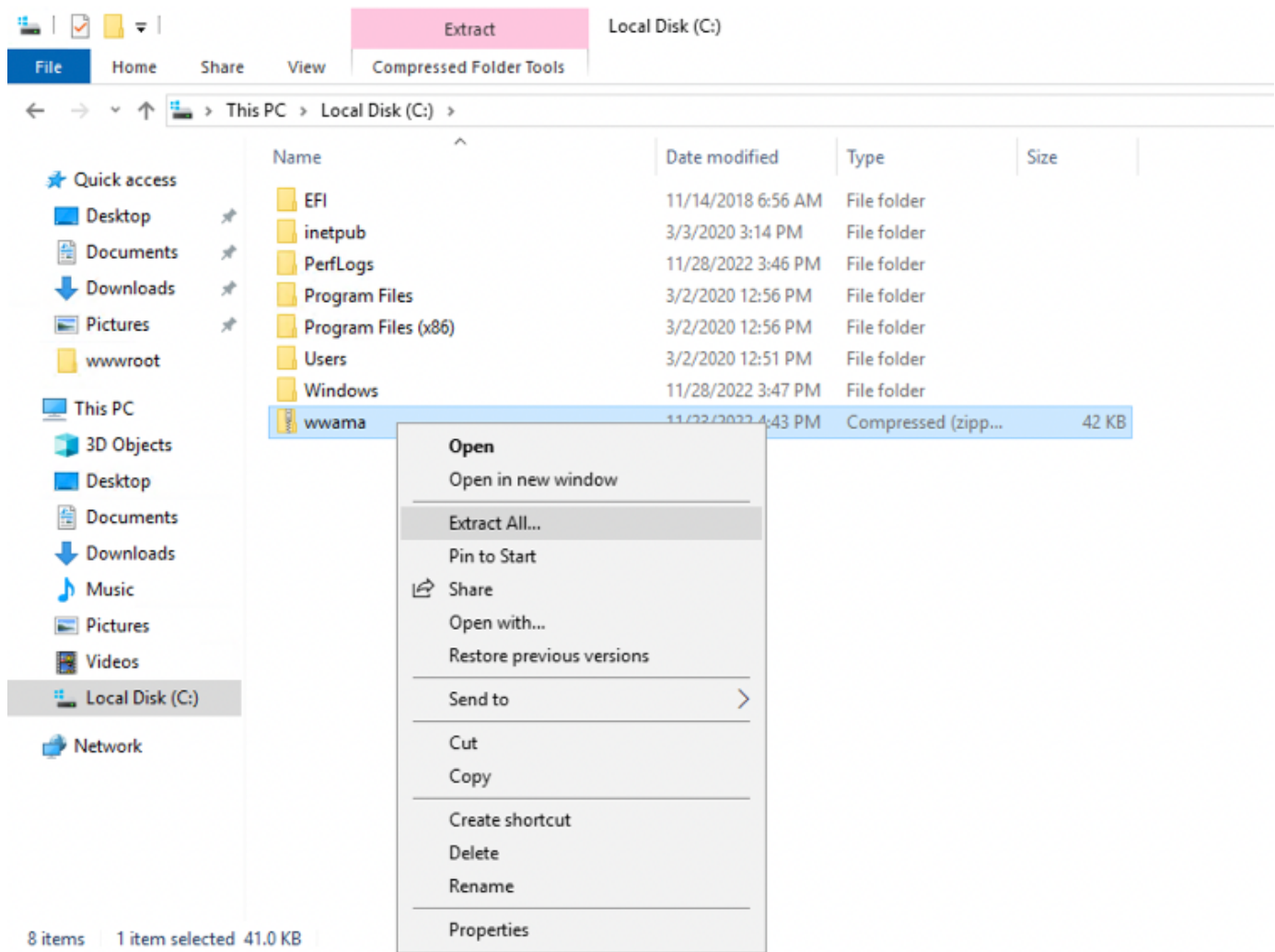
```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> Import-Module AWSPowerShell
PS C:\Users\Administrator> Set-AWSCredential -AccessKey [REDACTED] -SecretKey [REDACTED]
                        -StoreAs default
PS C:\Users\Administrator> 
```

7. Extract Migration Assistant files

The migration assistant has been pre-downloaded on the C:\ drive by the CloudFormation template. The file is **wwama.zip**.

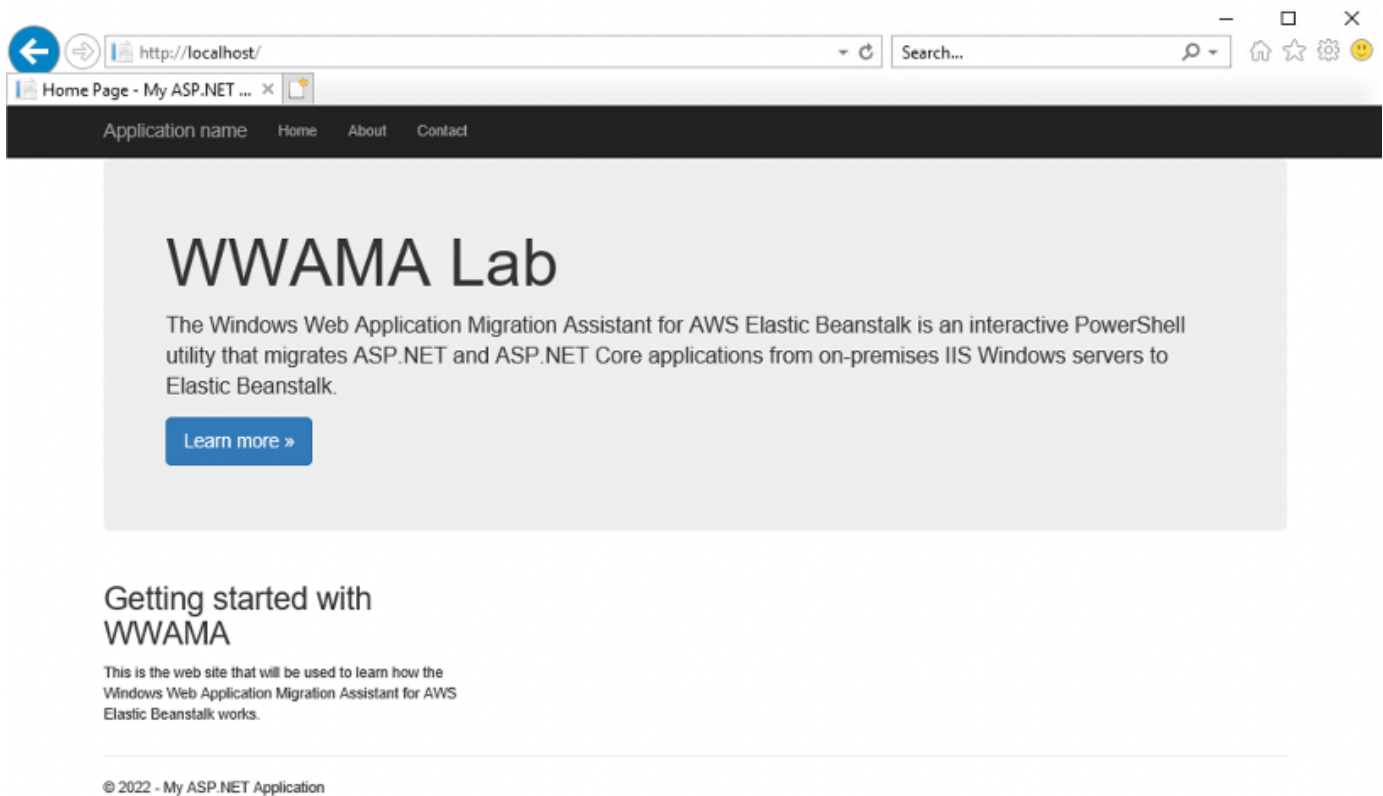
Open the context (right-click) menu for the wwama.zip file and extract the archive.



8. View the sample website

Open a web browser on the EC2 Windows Server instance and navigate to **<http://localhost/>**.

You will see the sample website that the migration assistant will migrate.



Step 5: Run the Migration Assistant

1. Launch the script

In the PowerShell terminal you opened earlier, change to the migration assistant directory and launch the migration script.

```
PS C:\> cd \wwama\windows-web-app-migration-assistant-master
PS C:\wwama\windows-web-app-migration-assistant-master> .
\MigrateIISWebsiteToElasticBeanstalk.ps1
```

The assistant prompts you for the location of your credentials file. Press **Enter** to skip.

When prompted for AWS Profile Name, enter **default**.

2. Select a Region

Enter the AWS Region where you'd like your Elastic Beanstalk environment to run.

For example: us-east-1. For a list of AWS Regions where Elastic Beanstalk is available, see [AWS Elastic Beanstalk endpoints and quotas](#).

3. Select the web application you want to migrate

The assistant then discovers any websites running on your IIS server and lists them, as in the example.

Enter the number **2** to migrate the sample site.

```
[AWS Migration] -----
[AWS Migration] | AWS Web Application Migration Assistant |
[AWS Migration] |                               v0.2 |
[AWS Migration] -----

[AWS Migration] Starting new migration run: run-20221206182200
[AWS Migration] Logs for this migration run can be found at
[AWS Migration]     C:\wwama\windows-web-app-migration-assistant-master\runs\run-20221206182200\logs
[AWS Migration] Checking for dependencies...
[AWS Migration] -----
[AWS Migration] Provide an AWS profile that the migrated application should use.
[AWS Migration] Enter file location, for example 'c:\aws\credentials', or press ENTER:
[AWS Migration] Enter the AWS profile name: default
[AWS Migration]
[AWS Migration] Enter the AWS Region for your migrated application. For example: us-east-2.
[AWS Migration] For a list of available AWS Regions, see:
[AWS Migration]     https://docs.aws.amazon.com/general/latest/gr/rande.html
[AWS Migration] Enter the AWS Region [us-east-1]:
[AWS Migration] -----
[AWS Migration] The migration assistant found website(s) on the local server 'EC2AMAZ-AEE0RST'.
[AWS Migration] [1] - Default Web Site
[AWS Migration] [2] - WWAMASite
[AWS Migration] Enter the number of the website to migrate: [1]: 2
[AWS Migration] Selected the website 'WWAMASite' to migrate from the server 'EC2AMAZ-AEE0RST'.
[AWS Migration] -----
```

4. Update connection strings

The assistant then prompts you to update any connection strings selected above. Press **Enter** as there aren't any connection strings in this application.

This message will appear: **The migration assistant didn't find any connection strings.**

5. Set up your Elastic Beanstalk application

Next, name your new Elastic Beanstalk application.

When prompted to select the Windows Server version, enter **6** and press **Enter**.

```

[AWS Migration] Taking a snapshot of the website... This might take a few minutes.
[AWS Migration] Automatically discovering possible connection strings... This might take a few minutes.
[AWS Migration] -----
[AWS Migration] The migration assistant didn't find any connection strings.
[AWS Migration] -----
[AWS Migration] Enter the number of the connection string you would like to update, or press ENTER:
[AWS Migration] Done. Collected 0 connection strings to update.
[AWS Migration] -----
[AWS Migration] Your application doesn't have a database that needs to be migrated.
[AWS Migration] Finished updating the connection string.
[AWS Migration] -----
[AWS Migration] Converting the website to an Elastic Beanstalk deployment bundle... This might take a few minutes.
[AWS Migration] An application bundle was successfully generated.
[AWS Migration] -----
[AWS Migration] AWS Elastic Beanstalk Deployment
[AWS Migration] -----
[AWS Migration] Enter a unique name for your new Elastic Beanstalk application: MigratedApp
[AWS Migration] Creating a new Elastic Beanstalk application...

ApplicationArn      : arn:aws:elasticbeanstalk:us-east-1:██████████:application/MigratedApp
ApplicationName     : MigratedApp
ConfigurationTemplates : {}
DateCreated         : 12/6/2022 6:23:25 PM
DateUpdated         : 12/6/2022 6:23:25 PM
Description          :
ResourceLifecycleConfig : Amazon.ElasticBeanstalk.Model.ApplicationResourceLifecycleConfig
Versions             : {}

[AWS Migration] Elastic Beanstalk supports the following Windows Server versions:
[AWS Migration] [1] : Windows Server 2012
[AWS Migration] [2] : Windows Server 2012 R2
[AWS Migration] [3] : Windows Server Core 2012 R2
[AWS Migration] [4] : Windows Server 2016
[AWS Migration] [5] : Windows Server Core 2016
[AWS Migration] [6] : Windows Server 2019
[AWS Migration] [7] : Windows Server Core 2019
[AWS Migration] Enter the number of the Windows version for your Elastic Beanstalk environment [1]: 6

```

6. Configure application

For instance type that the application will run on, press **Enter** to select the default (t3.medium).

Enter the instance type (default t3.medium) :

For **environment type**, enter 1 for single instance.

Enter the environment type [1]:

The migration assistant then migrates your application to Elastic Beanstalk.

```

[AWS Migration] Taking a snapshot of the website... This might take a few minutes.
[AWS Migration]
[AWS Migration] Automatically discovering possible connection strings... This might take a few minutes.
[AWS Migration] -----
[AWS Migration] The migration assistant didn't find any connection strings.
[AWS Migration] -----
[AWS Migration] Enter the number of the connection string you would like to update, or press ENTER:
[AWS Migration]
[AWS Migration] Done. Collected 0 connection strings to update.
[AWS Migration] -----
[AWS Migration] Your application doesn't have a database that needs to be migrated.
[AWS Migration] Finished updating the connection string.
[AWS Migration] -----
[AWS Migration] Converting the website to an Elastic Beanstalk deployment bundle... This might take a few minutes.
[AWS Migration] An application bundle was successfully generated.
[AWS Migration] -----
[AWS Migration] AWS Elastic Beanstalk Deployment
[AWS Migration] -----
[AWS Migration] Enter a unique name for your new Elastic Beanstalk application: MigratedApp
[AWS Migration] Creating a new Elastic Beanstalk application...

ApplicationArn      : arn:aws:elasticbeanstalk:us-east-1:██████████:application/MigratedApp
ApplicationName     : MigratedApp
ConfigurationTemplates : {}
DateCreated        : 12/6/2022 6:23:25 PM
DateUpdated        : 12/6/2022 6:23:25 PM
Description         :
ResourceLifecycleConfig : Amazon.ElasticBeanstalk.Model.ApplicationResourceLifecycleConfig
Versions           : {}

[AWS Migration] Elastic Beanstalk supports the following Windows Server versions:
[AWS Migration] [1] : Windows Server 2012
[AWS Migration] [2] : Windows Server 2012 R2
[AWS Migration] [3] : Windows Server Core 2012 R2
[AWS Migration] [4] : Windows Server 2016
[AWS Migration] [5] : Windows Server Core 2016
[AWS Migration] [6] : Windows Server 2019
[AWS Migration] [7] : Windows Server Core 2019
[AWS Migration] Enter the number of the Windows version for your Elastic Beanstalk environment [1]: 6

```

7. Verify completion

When the migration completes, you will see a success message in the CLI.

```
[AWS Migration] Updating the Elastic Beanstalk environment... This might take a few minutes.
.....
LoggedAt                : 12/6/2022 6:30:42 PM
AbortableOperationInProgress : True
ApplicationName          : MigratedApp
CNAME                    : MigratedApp-env.eba-eqgwqiqh.us-east-1.elasticbeanstalk.com
DateCreated              : 12/6/2022 6:23:35 PM
DateUpdated              : 12/6/2022 6:30:41 PM
Description               :
EndpointURL              : 3.231.48.118
EnvironmentArn            : arn:aws:elasticbeanstalk:us-east-1: :environment/MigratedApp/MigratedApp-env
EnvironmentId             : e-8caujwcefr
EnvironmentLinks          : {}
EnvironmentName           : MigratedApp-env
Health                   : Grey
HealthStatus              :
PlatformArn              : arn:aws:elasticbeanstalk:us-east-1::platform/IIS 10.0 running on 64bit Windows Server
                           2019/2.10.6
Resources                 :
SolutionStackName         : 64bit Windows Server 2019 v2.10.6 running IIS 10.0
Status                    : Updating
TemplateName              :
Tier                      : Amazon.ElasticBeanstalk.Model.EnvironmentTier
VersionLabel              : run-20221206182200-v1
ResponseMetadata           : Amazon.Runtime.ResponseMetadata
ContentLength              : 1325
HttpStatusCode             : OK

[AWS Migration] Waiting for the Elastic Beanstalk application to launch... This might take a few minutes.
.
[AWS Migration] Note that it might take a few minutes for the application to be ready.
[AWS Migration] Application URL:
[AWS Migration]     MigratedApp-env.eba-eqgwqiqh.us-east-1.elasticbeanstalk.com
[AWS Migration]
[AWS Migration] The Elastic Beanstalk deployment succeeded. Your web application is now hosted in AWS at 'MigratedApp-env.eba-eqgwqiqh.us-east-1.elasticbeanstalk.com'

PS C:\wwama\windows-web-app-migration-assistant-master>
```

Step 6: Navigate to your web application hosted on Elastic Beanstalk

Now that the site is successfully migrated, verify that the website is up and running.

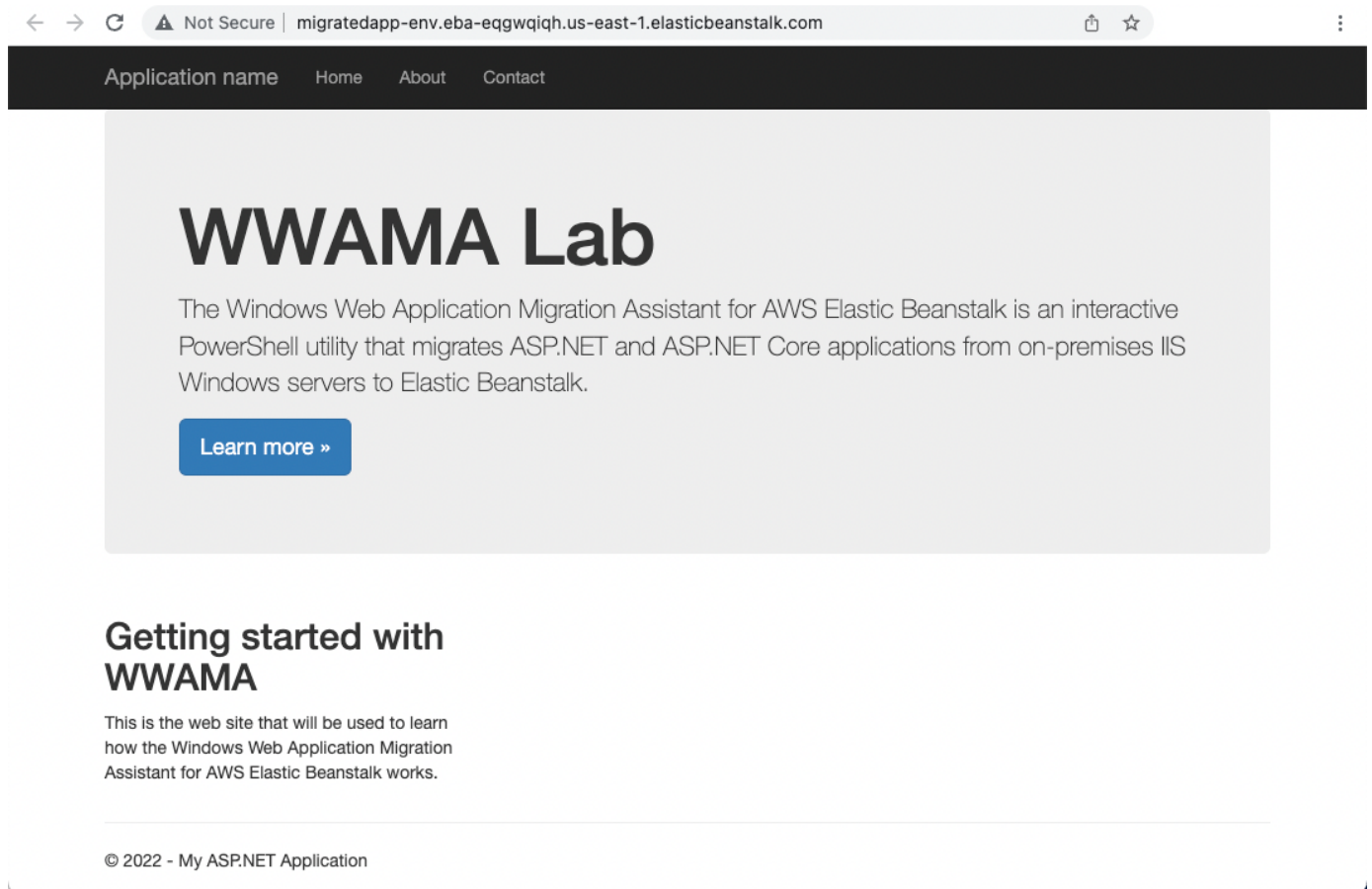
1. Locate the application URL

Get the URL from the output of the PowerShell script.

```
[AWS Migration] Note that it might take a few minutes for the application to be ready.
[AWS Migration] Application URL:
[AWS Migration]     MigratedApp-env.eba-eqgwqiqh.us-east-1.elasticbeanstalk.com
[AWS Migration]
[AWS Migration] The Elastic Beanstalk deployment succeeded. Your web application is now hosted in AWS at 'MigratedApp-env.eba-eqgwqiqh.us-east-1.elasticbeanstalk.com'
```

2. View the page

Input the URL into your web browser, and you should see your web application, now running on Elastic Beanstalk.



3. Access the app in Beanstalk

You can also view the Elastic Beanstalk environment from the [AWS console](#). Make sure you're seeing the console for the same Region you deployed your application to. Feel free to explore what you can do with your application by using the menu on the left side.

The screenshot shows the Elastic Beanstalk console with the 'MigratedApp-env' environment selected. The left sidebar shows the navigation menu with 'Environments' selected. The main content area displays the environment details for 'MigratedApp-env' (Application name: MigratedApp). The 'Health' section shows a green checkmark and 'Ok' status. The 'Running version' section shows 'run-20221206182200-v1' and an 'Upload and deploy' button. The 'Platform' section shows 'IIS 10.0 running on 64bit Windows Server 2019/2.10.6' and a 'Change' button. Below these sections is a 'Recent events' table with 5 entries, all of type 'INFO'.

Time	Type	Details
2022-12-06 13:32:05 UTC-0500	INFO	Environment update completed successfully.
2022-12-06 13:32:05 UTC-0500	INFO	New application version was deployed to running EC2 instances.
2022-12-06 13:30:48 UTC-0500	INFO	Deploying new version to instance(s).
2022-12-06 13:30:41 UTC-0500	INFO	Environment update is starting.
2022-12-06 13:30:40 UTC-0500	INFO	Environment health has transitioned from Pending to Ok. Initialization completed 10 seconds ago and took 7 minutes.

Clean up resources

1. Delete the Elastic Beanstalk application

Go to the [Elastic Beanstalk console](#).

In the **Applications** view, select the radio button next to your application, choose the **Actions** menu, then select **Delete application**. In the confirmation dialog, enter the name of your application and choose **Delete**. This will delete both your application and the Elastic Beanstalk environment.

The screenshot shows the Elastic Beanstalk console with the 'Applications' view selected. The left sidebar shows the navigation menu with 'Applications' selected. The main content area displays 'All applications' with a search bar and a table of applications. The 'MigratedApp' application is selected, and the 'Actions' menu is open, showing options like 'Create environment', 'Delete application', 'View application versions', 'View saved configurations', and 'Restore terminated environment'. The 'Delete application' option is highlighted with a red box.

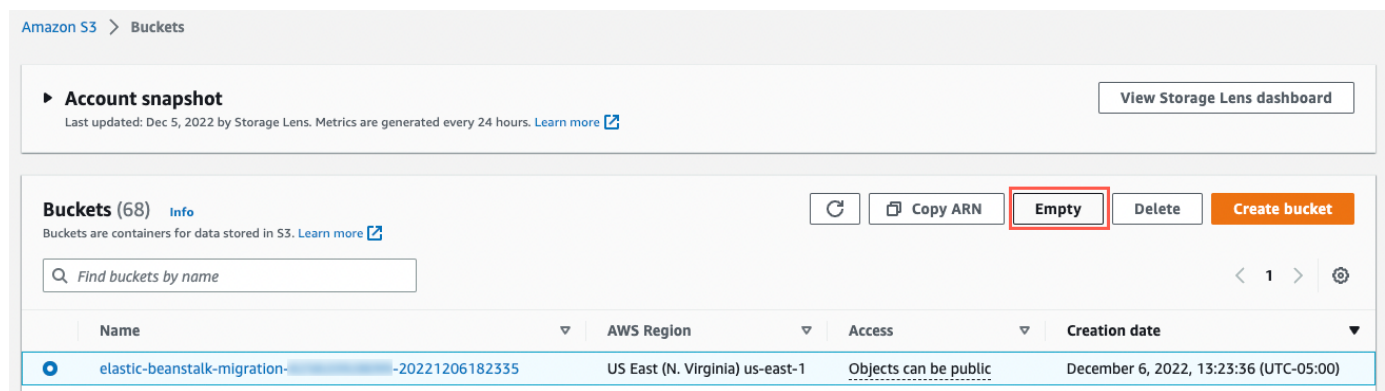
Application name	Environments	Date created	Last modified	ARN
MigratedApp	MigratedApp-env MigratedApp-env	2022-12-06 13:23:25 UTC-0500	2022-12-06 13:23:25 UTC-0500	arn:aws:elasticbeanstalk:us-east-1:825820928099:application/MigratedApp

2. Delete temporary files from Amazon S3

Go to the [Amazon S3 console](#), select the radio button next to the bucket named **elastic-beanstalk-migration-ACCOUNT_ID-TIMESTAMP** **Amazon Managed Service for Prometheus**, where **ACCOUNT_ID** is your AWS account ID and **TIMESTAMP** **Amazon Managed Service for Prometheus** is the time that your application was deployed.

Then choose **Empty** to first empty the contents of the bucket. You will need to confirm this action by entering **permanently delete** in a text box and choose **Empty**. Choose **Exit** to go back to the S3 buckets view.

Now that the S3 bucket is empty, select the radio button next to the bucket again and choose **Delete**. You will need to confirm this action by entering the name of the bucket in the text box and choose **Delete bucket**.



3. Delete the CloudFormation stack

Go to the CloudFormation console and delete the [CloudFormation stack](#) **WWAMASStack** created at the start of the lab.

Conclusion

Congratulations! You have successfully migrated a sample ASP.NET web application to a fully managed Elastic Beanstalk environment using the Windows Web Application Migration Assistant (WWAMA). AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS. You can simply upload your code, and Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, auto scaling, to application health monitoring. At the same time, you retain full control over the AWS resources powering your application and can access the underlying resources at any time. Visit [AWS Elastic Beanstalk](#) to learn more.