



Prakash Rao

Posted on Jun 18, 2024 • Edited on Mar 28



Deploying a "Hello World" Application to AWS Elastic Beanstalk

Introduction

AWS Elastic Beanstalk as 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. This post will demonstrate how to deploy your application in AWS Elastic Beanstalk.

Prerequisites

- AWS account
- Basic knowledge of web application development
- Familiarity with AWS services is helpful but not required

Step 1: Creating a Simple Web Application

Initial Setup

- Choose a programming language and framework. For this example, let's use Python with Flask.

- Create a new directory for your project and navigate into it.
- Initialize a new Python virtual environment and activate it (optional but recommended).

Application Code

- Create a file named application.py and add the following Flask application code:

```
from flask import Flask

application = Flask(__name__)

@application.route('/')
def hello_world():
    return 'Hello, World!'

if __name__ == '__main__':
    application.run()
```

- Create a requirements.txt file specifying Flask:

```
Flask==1.1.2
```

Local Testing (Optional)

- Run the application locally to ensure it works.
- Open a browser and navigate to <http://localhost:5000> to see the "Hello, World!" message.

Step 2: Preparing the Application for Deployment

- Zip the application.py and requirements.txt files together.

```
zip myapp.zip application.py requirements.txt
```

Step 3: Creating an Elastic Beanstalk Environment

- Log in to the AWS Management Console.
- Navigate to the Elastic Beanstalk service and click "Create New Application".
- Enter an application name and description.
- Create a new environment within this application – choose the Web server environment.

Configure environment Info

Environment tier Info

Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.

Web server environment

Run a website, web application, or web API that serves HTTP requests. [Learn more](#) 

Worker environment

Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. [Learn more](#) 

Application information Info

Application name

Hello World App

Maximum length of 100 characters.

► **Application tags (optional)**

- Select the Python platform and choose the appropriate version.

Platform Info

Platform type

Managed platform

Platforms published and maintained by Amazon Elastic Beanstalk. [Learn more](#) 

Custom platform

Platforms created and owned by you. This option is unavailable if you have no platforms.

Platform

Python



Platform branch

Python 3.11 running on 64bit Amazon Linux 2023



Platform version

4.1.0 (Recommended)



Step 4: Uploading and Deploying the Application

- When prompted to upload your code, choose the "Upload your code" option and upload the myapp.zip file.

Application code Info

Sample application

Existing version

Application versions that you have uploaded.

Upload your code

Upload a source bundle from your computer or copy one from Amazon S3.

Version label

Unique name for this version of your application code.

1

Source code origin. Maximum size 500 MB

Local file

Upload application

 Choose file

 File name: myapp.zip

File must be less than 500MB max file size

Public S3 URL

- Configure more options if needed, or simply click "Create environment".

Presets Info

Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.

Configuration presets

- Single instance (free tier eligible)
- Single instance (using spot instance)
- High availability
- High availability (using spot and on-demand instances)
- Custom configuration

Step 5: Configuring Service Access

- For Elastic Beanstalk selecting role is crucial. For this lab purpose let's chose existing service role and existing EC2 instance profile.

Configure service access Info

Service access

IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. [Learn more](#)

Service role

- Create and use new service role
 Use an existing service role

Existing service roles

Choose an existing IAM role for Elastic Beanstalk to assume as a service role. The existing IAM role must have the required IAM managed policies.



EC2 key pair

Select an EC2 key pair to securely log in to your EC2 instances. [Learn more](#)



EC2 instance profile

Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.

[View permission details](#)[Cancel](#)[Skip to review](#)[Previous](#)[Next](#)

Step 6: Environment Configuration and Launch

- For rest of the optional parameters, keep it as is.

Step 1

[Configure environment](#)

Step 2

[Configure service access](#)

Step 3 - *optional*

[Set up networking, database, and tags](#)

Step 4 - *optional*

[Configure instance traffic and scaling](#)

Step 5 - *optional*

[Configure updates, monitoring, and logging](#)

Step 6

Review

- AWS Elastic Beanstalk will now create your environment and deploy the application.

[Cancel](#)

[Previous](#)

[Submit](#)

- This process might take a few minutes.

Elastic Beanstalk is launching your environment. This will take a few minutes.

Elastic Beanstalk > Environments > Hello-world-env-1

Hello-world-env-1 [Info](#)

Events | Health | Logs | Monitoring | Alarms | Managed updates | Tags

Environment overview

| | | | |
|--------|---|------------------|--------------|
| Health | Pending | Environment ID | e-3uzrggmmug |
| Domain | Hello-world-env-1.eba-mk4j73ri.us-east-1.elasticbeanstalk.com | Application name | hello-world |

Platform

| | | | |
|-----------------|--|----------------|-----------|
| Platform | Python 3.11 running on 64bit Amazon Linux 2023/4.1.0 | | |
| Running version | 1 | Platform state | Supported |

Events (8) [Info](#)

| Time | Type | Details |
|--------------------------------|------|---|
| June 18, 2024 07:47:34 (UTC+9) | INFO | Instance deployment completed successfully. |
| June 18, 2024 07:47:30 (UTC+9) | INFO | Instance deployment successfully generated a 'Procfile'. |
| June 18, 2024 07:46:48 (UTC+9) | INFO | Waiting for EC2 instances to launch. This may take a few minutes. |

Step 7: Accessing the Deployed Application

- Once the environment is ready, click the provided URL to access your application.

Environment successfully launched.

Elastic Beanstalk > Environments > Hello-world-env-1

Hello-world-env-1 [Info](#)

Events | Health | Logs | Monitoring | Alarms | Managed updates | Tags

Environment overview

| | | | |
|--------|---|------------------|--------------|
| Health | Pending | Environment ID | e-3uzrggmmug |
| Domain | Hello-world-env-1.eba-mk4j73ri.us-east-1.elasticbeanstalk.com | Application name | hello-world |

Platform

| | | | |
|-----------------|--|----------------|-----------|
| Platform | Python 3.11 running on 64bit Amazon Linux 2023/4.1.0 | | |
| Running version | 1 | Platform state | Supported |

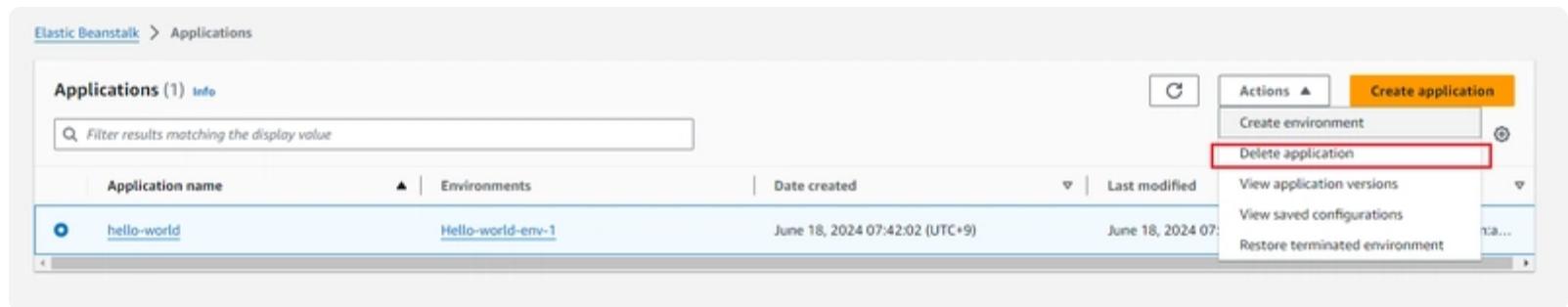
- You should see the "Hello, World!" message served from AWS Elastic Beanstalk.



Hello, World!

Step 8: Clean Up

- To avoid incurring charges, delete the Elastic Beanstalk environment and application.



Conclusion

Congratulations! You've just taken a significant step into the world of web application deployment using AWS Elastic Beanstalk. By following the steps outlined in this post, you have learned how to deploy a "Hello World" application.

As you become more comfortable with AWS Elastic Beanstalk, I encourage you to explore its advanced features, experiment with different configurations, and consider how you can integrate other AWS services to enhance your application's functionality and performance.