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.

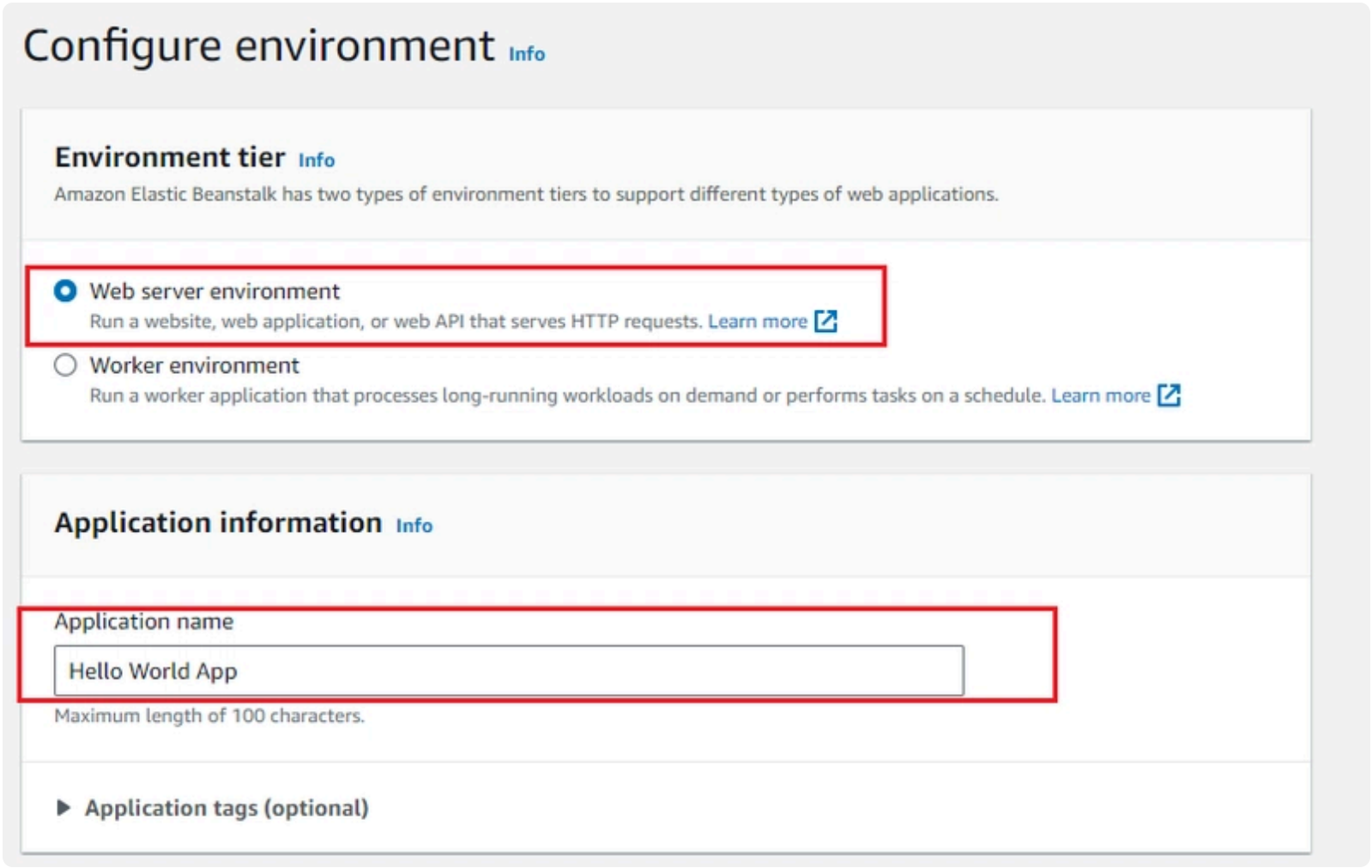
Step 2: Preparing the Application for Deployment

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

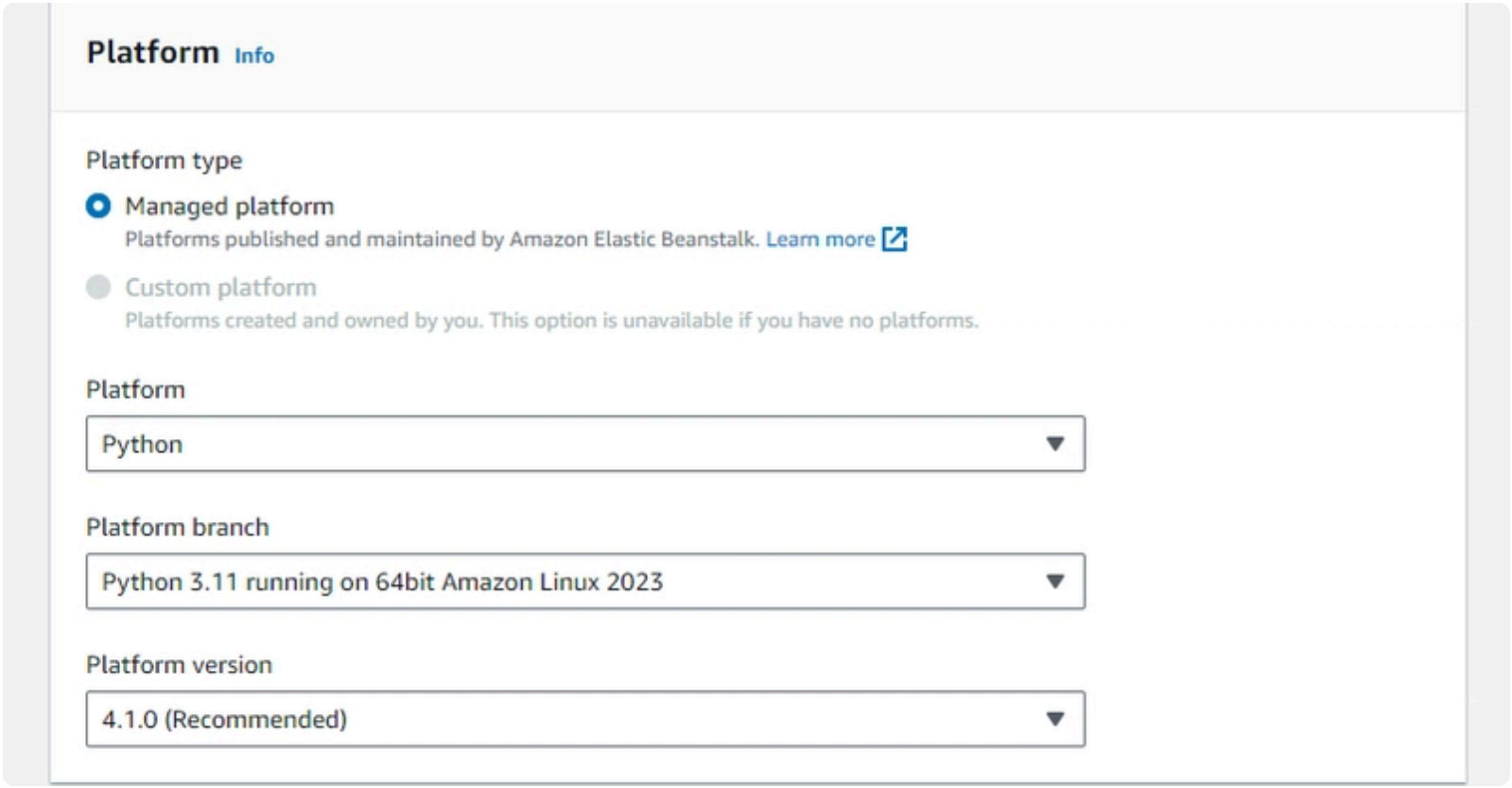
```
zip myapp.zip application.py requirements.txt
```

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.

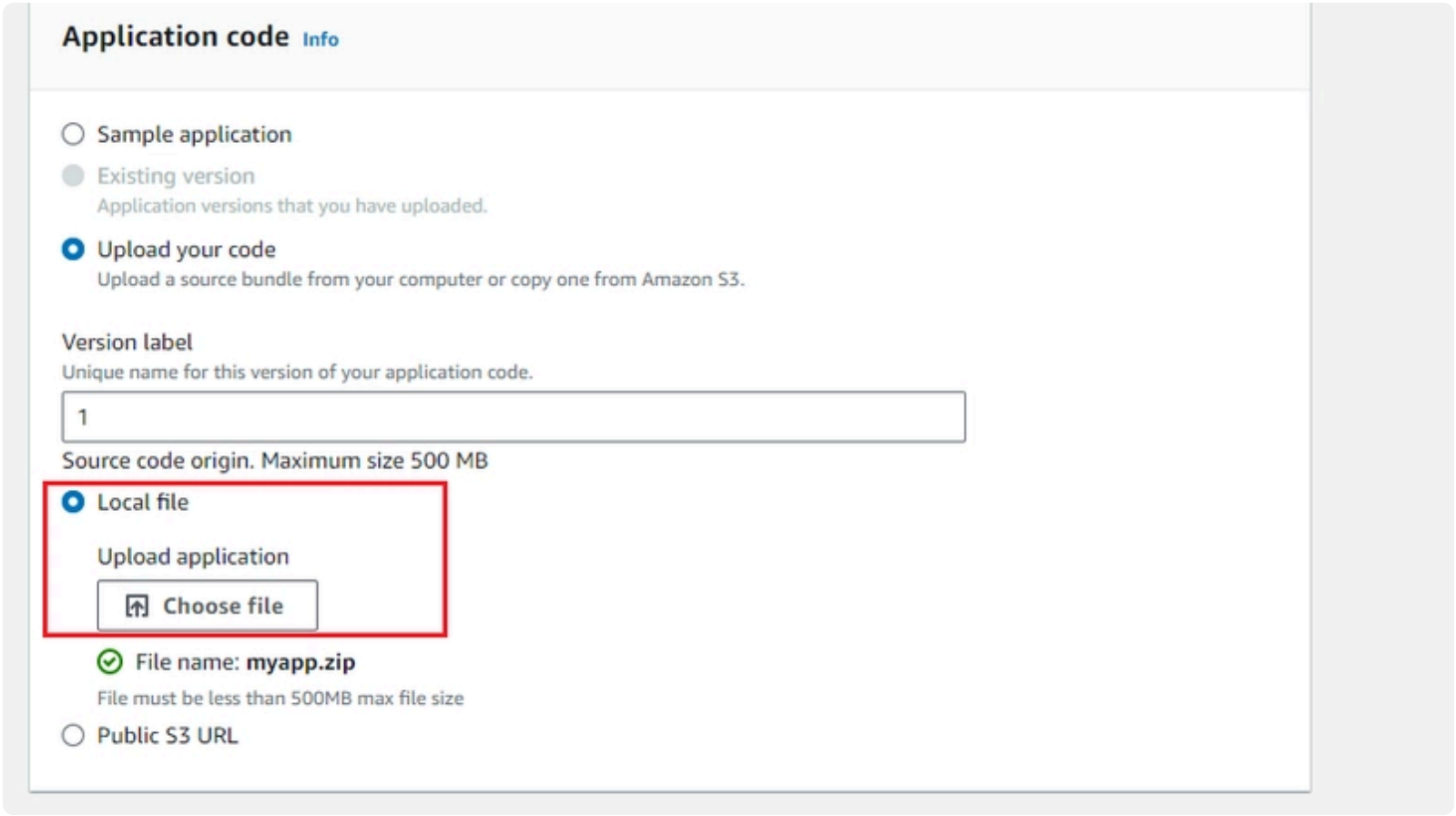


- Select the Python platform and choose the appropriate version.

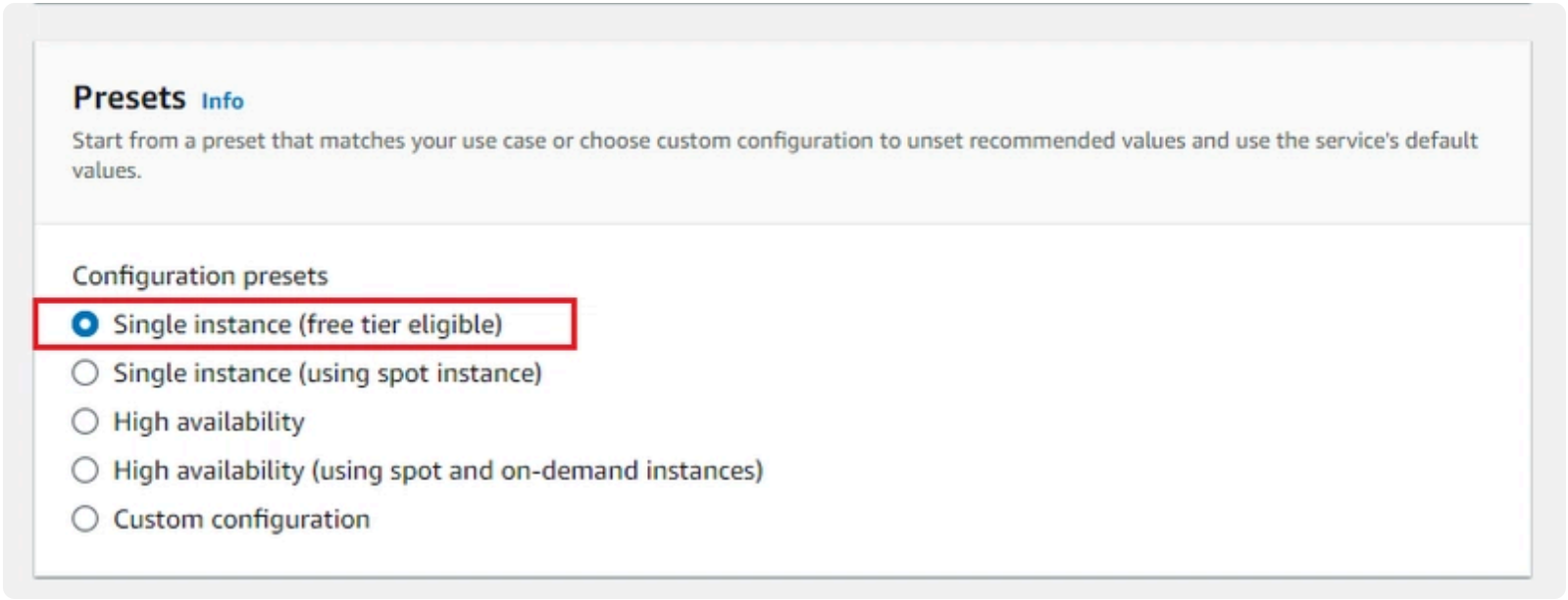


Step 4: Uploading and Deploying the Application

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

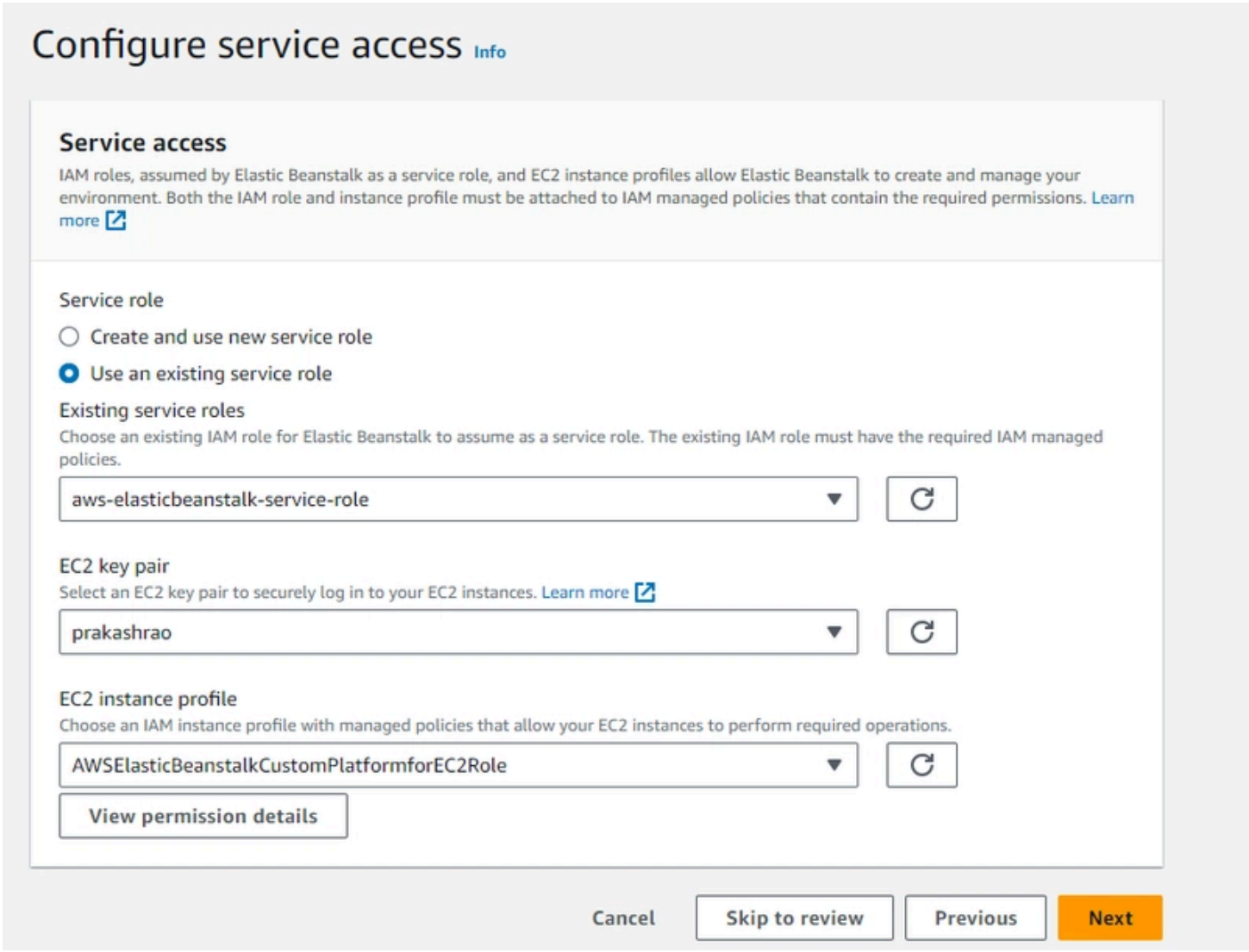


- Configure more options if needed, or simply click "Create environment".



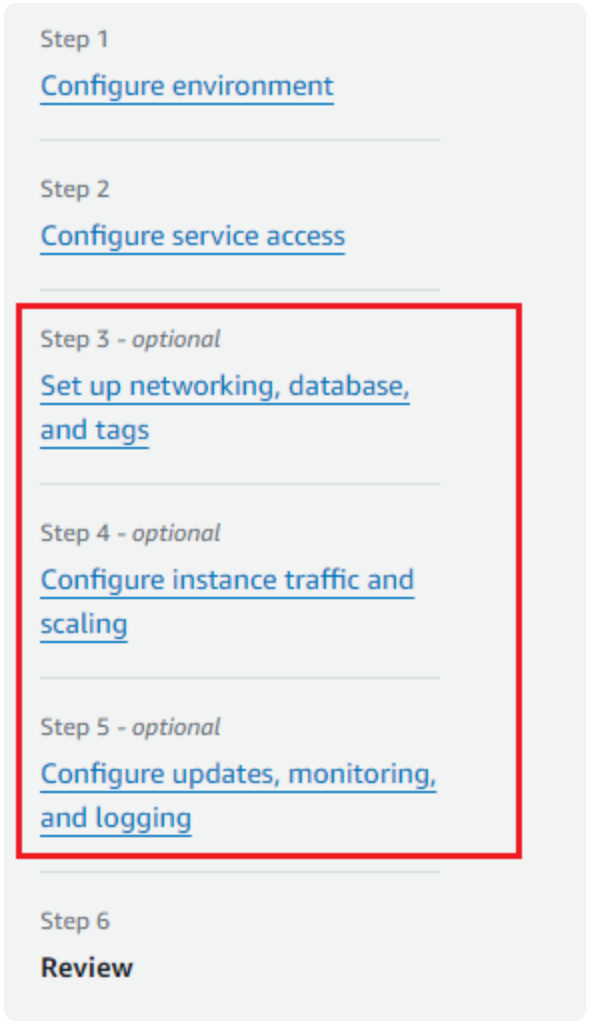
Step 5: Configuring Service Access

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

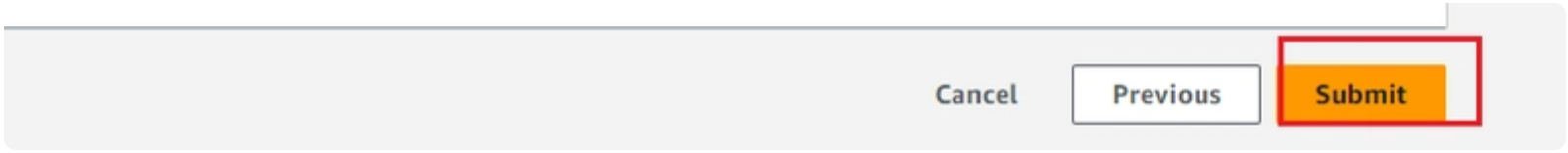


Step 6: Environment Configuration and Launch

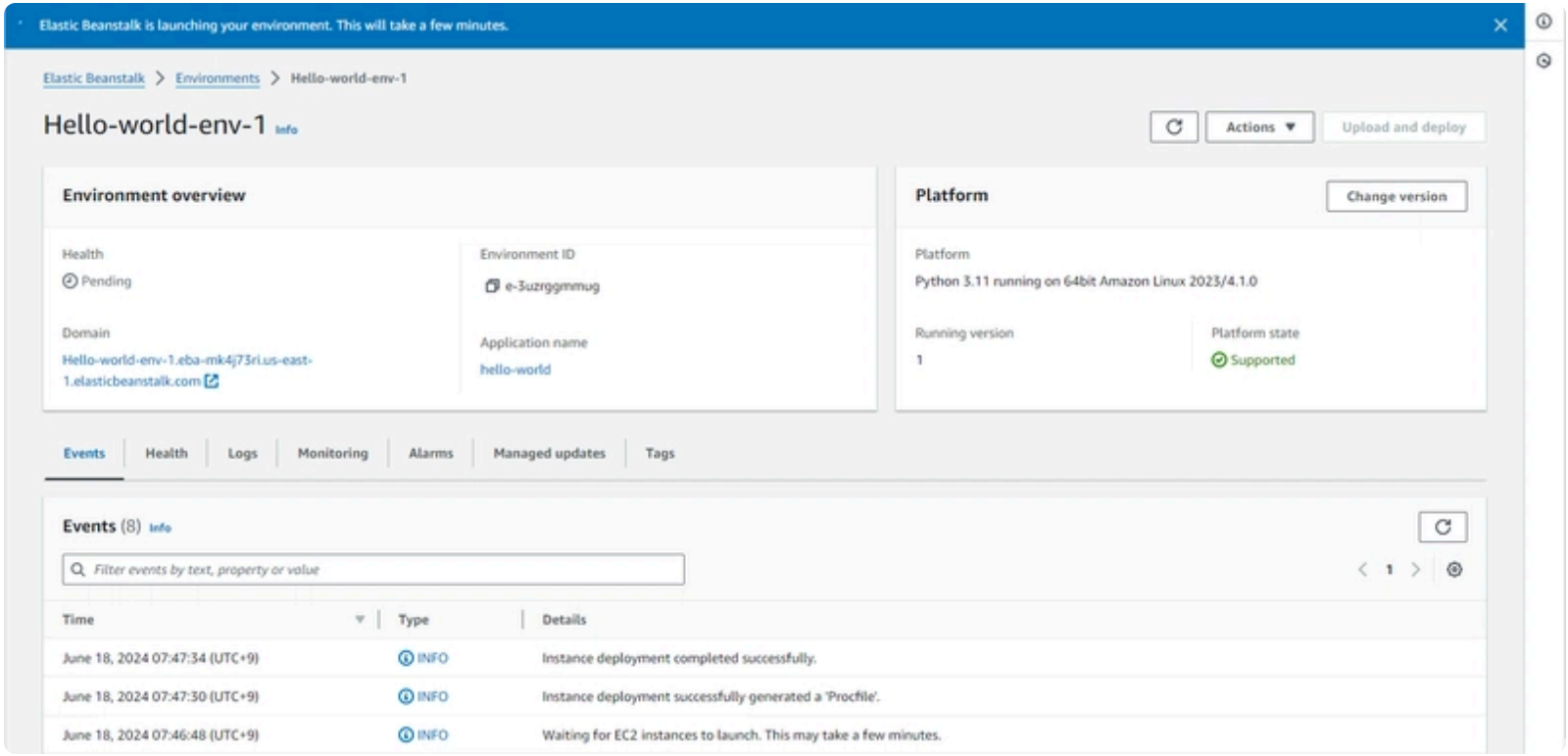
- For rest of the optional parameters, keep it as is.



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

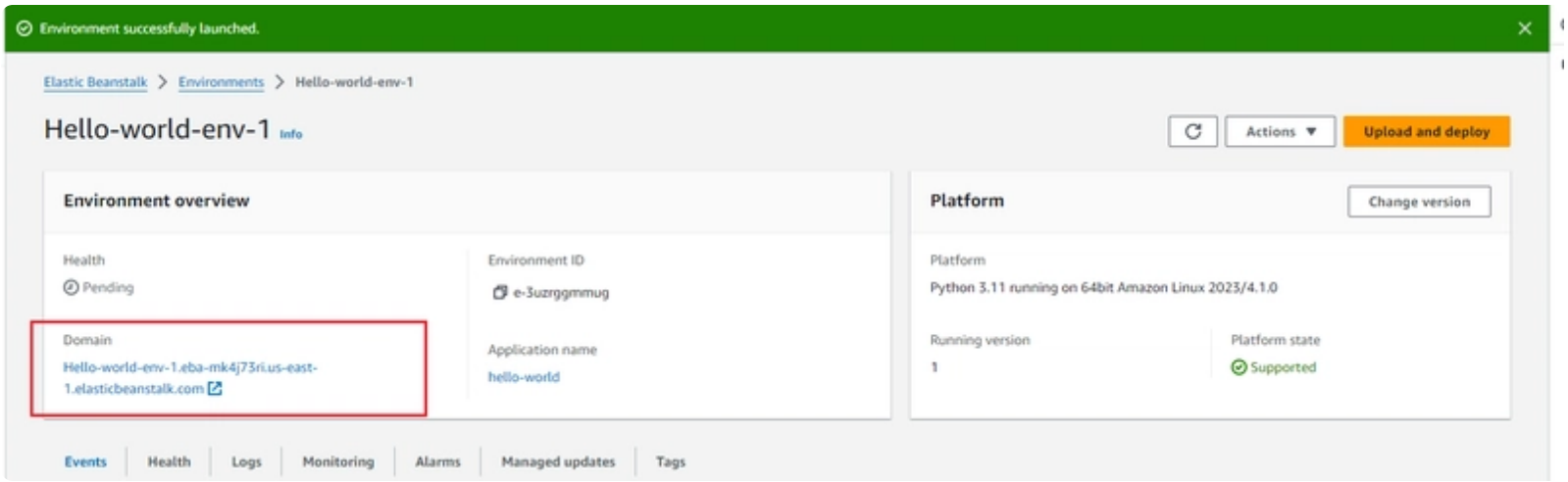


- This process might take a few minutes.

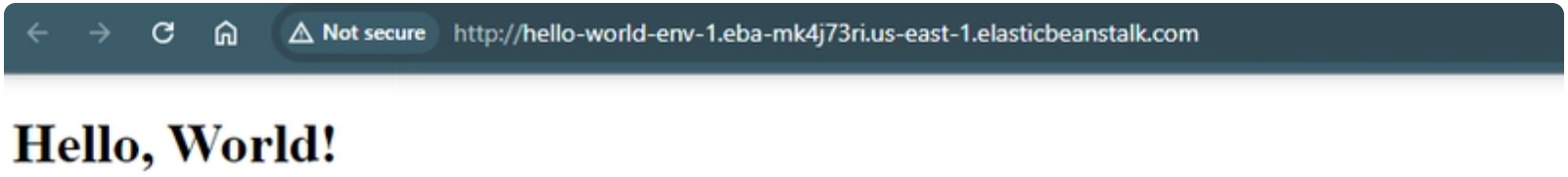


Step 7: Accessing the Deployed Application

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

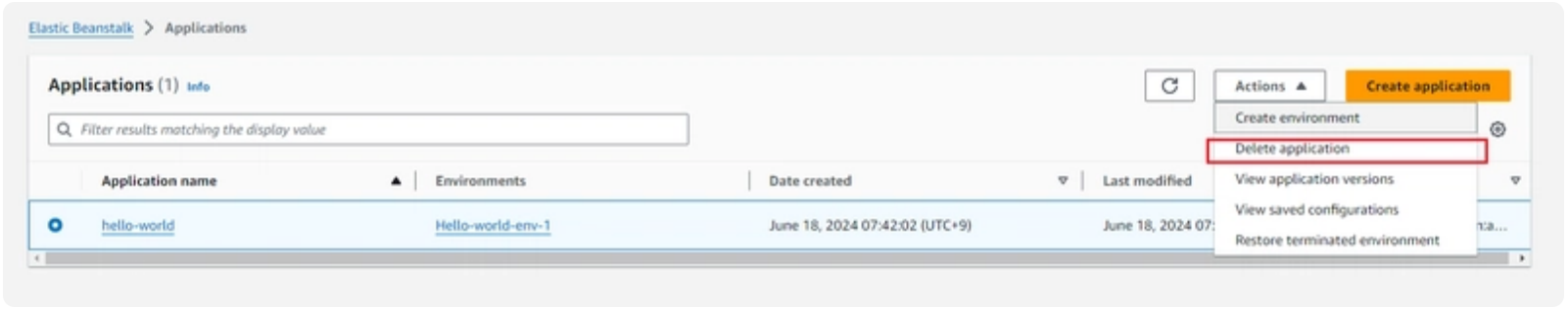


- You should see the "Hello, World!" message served from AWS Elastic Beanstalk.



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.