

Module 4: Application Deployment Using Elastic Beanstalk

Demo Document 2

edureka!

© Brain4ce Education Solutions Pvt. Ltd.

Deploy An Application In Beanstalk Using Docker

Demo steps:

Step 1: Create an Application

- In your local system, create a file with the name application.py
- Type the below code and save it local system in the name of **application.py**

```
from flask import Flask

# Print a nice greeting

def say_hello(username = "World"):

    return '<html><body background="https://bit.ly/2NuGl9Q" background-  
position=center background-repeat=no-repeat background-size=cover style="padding:  
210px 0; background-color:#000" ><font color="white"><center><h1>Hello  
%s!</h1></font><br><br><br></body>' % username

# Some bits of text for the page

header_text = '''

<html>\n<head> <title> Docker Demo</title> </head>\n<body>'''

instructions = '''

<font color="white"><h2><em> Hey!!!! It is Working </h2></font>\n'''

home_link = '<p><a href="/">Back</a></p>\n'

footer_text = '</body>\n</html>'

# Elastic Beanstalk looks for an 'application' that is callable by default

application = Flask(__name__)

# Add a rule for the index page

application.add_url_rule('/', 'index', (lambda: header_text +  
    say_hello() + instructions + footer_text))
```

```
# Add a rule when the page is accessed with a name appended to the site

# URL

application.add_url_rule('/<username>', 'hello', (lambda username:
    header_text + say_hello(username) + home_link + footer_text))

# Run the application

if __name__ == "__main__":
    # Setting debug to True enables debug output. This line should be
    # removed before deploying a production application.

    application.debug = True

    application.run(host="0.0.0.0")
```

Step 2: Create a Dockerfile


- In your Notepad, type the below code
- And Save it in the name of Dockerfile

```
FROM python:3.6
COPY . /app
WORKDIR /app
RUN pip install Flask==1.0.2
EXPOSE 5000
CMD ["python", "application.py"]
```

Step 3: Create a source bundle

- Zip the Dockerfile and the application.py file to form the source bundle

Step 4: Create a new single docker environment in beanstalk and upload the myapp.zip bundle



Create a web server environment

Launch an environment with a sample application or your own code. By creating an environment, you allow AWS Elastic Beanstalk to manage AWS resources and permissions on your behalf. [Learn more](#)

Environment information

Choose the name, subdomain, and description for your environment. These cannot be changed later.

Application name single container

Environment name

Domain

Description

Base configuration

Platform ☒ Preconfigured platform
Platforms published and maintained by AWS Elastic Beanstalk.

☐ Custom platform
Platforms created and owned by you. [Learn more](#)

Application code ☐ Sample application
Get started right away with sample code.

☐ Existing version
Application versions that you have uploaded for single container.

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

ZIP or WAR

Upload your code

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

Source code origin ☒ Local file
(Maximum size 512 MB)

myapp.zip

☐ Public S3 URL

Version label

Unique name for this version of your application code.

Step 5: Once the environment is Created, test it with the URL if the application works

