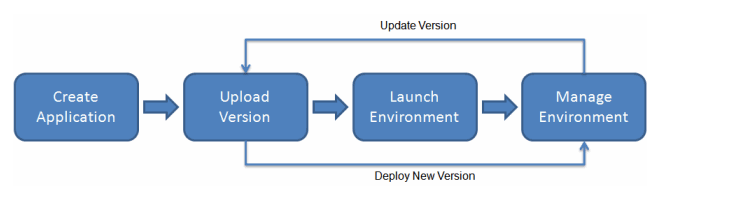
<https://aws.amazon.com/whitepapers/>

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/GettingStarted.html>

With Elastic Beanstalk, you can quickly deploy and manage applications in the AWS Cloud without worrying about the infrastructure that runs those applications. AWS Elastic Beanstalk reduces management complexity without restricting choice or control. You simply upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring

lastic Beanstalk supports applications developed in Go, Java, .NET, Node.js, PHP, Python, and Ruby,



AWS Elastic Beanstalk **lets you manage all of the resources that run your application as environments.**

## Storage

Elastic Beanstalk does not restrict your choice of persistent storage and database service options

## Pricing

There is no additional charge for Elastic Beanstalk. You pay only for the underlying AWS resources that your application consumes.

## Application

An Elastic Beanstalk application is a logical collection of Elastic Beanstalk components, including environments, versions, and environment configurations. In Elastic Beanstalk an application is conceptually similar to a folder.

## Application Version

In Elastic Beanstalk, an application version refers to a specific, labeled iteration of deployable code for a web application. An application version points to an Amazon Simple Storage Service (Amazon S3) object that contains the deployable code such as a Java WAR file.

## Environment

An environment is a version that is deployed onto AWS resources.

## Environment Tier

The environment tier that you choose determines whether Elastic Beanstalk provisions resources to support an application that handles HTTP requests or an application that pulls tasks from a queue. An application that serves HTTP requests runs in a [web server environment](https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/concepts-webserver.html). An environment that pulls tasks from an Amazon Simple Queue Service queue runs in a [worker environment](https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/concepts-worker.html).

## Environment Configuration

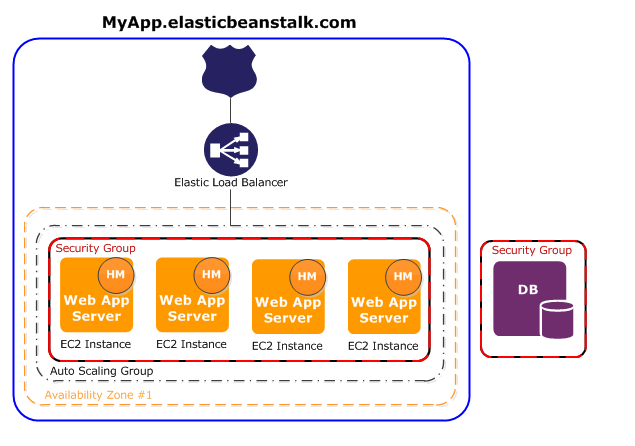
An environment configuration identifies a collection of parameters and settings that define how an environment and its associated resources behave. When you update an environment’s configuration settings, Elastic Beanstalk automatically applies the changes to existing resources or deletes and deploys new resources (depending on the type of change).

## Configuration Template

A configuration template is a starting point for creating unique environment configurations.

# Web Server Environments

AWS resources created for an environment include one elastic load balancer (ELB in the diagram), an Auto Scaling group, and one or more Amazon EC2 instances.



Every environment has a CNAME (URL) that points to a load balancer. The environment has a URL such as myapp.us-west-2.elasticbeanstalk.com. This URL is aliased in [Amazon Route 53](https://aws.amazon.com/route53/) to an Elastic Load Balancing URL—something like abcdef-123456.us-west-2.elb.amazonaws.com—by using a CNAME record. [Amazon Route 53](https://aws.amazon.com/route53/) is a highly available and scalable Domain Name System (DNS) web service.

 In addition, a software component called the *host manager (HM)* runs on each Amazon EC2 server instance. (In the diagram, the HM is an orange circle in each EC2 instance.) The host manager is responsible for:

* Deploying the application
* Aggregating events and metrics for retrieval via the console, the API, or the command line
* Generating instance-level events
* Monitoring the application log files for critical errors
* Monitoring the application server
* Patching instance components
* Rotating your application's log files and publishing them to Amazon S3

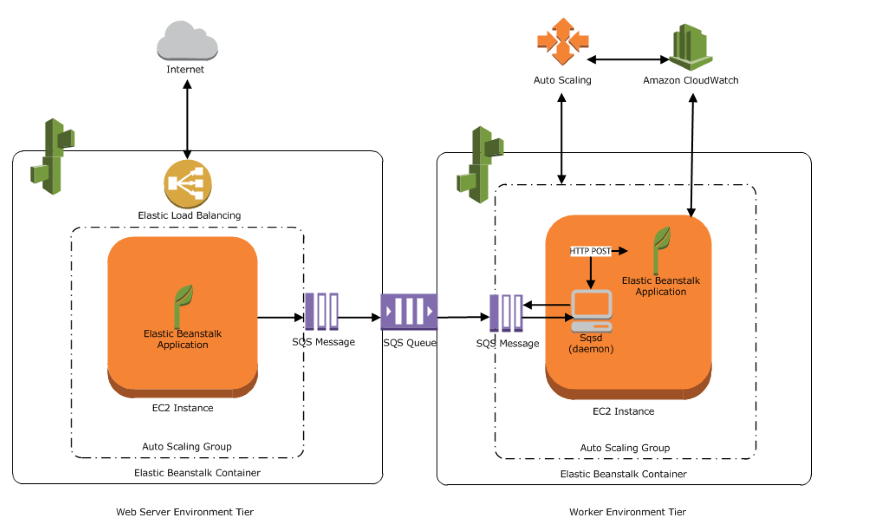
**Worker Environments**

AWS resources created for a worker environment tier include an

* Auto Scaling group,
* one or more Amazon EC2 instances,
* IAM role.
* SQS Queue

When you launch a worker environment tier, Elastic Beanstalk installs the necessary support files for your programming language of choice and a **daemon on each EC2 instance** in the Auto Scaling group.

The daemon is responsible for pulling requests from an Amazon SQS queue and then sending the data to the web application running in the worker environment tier that will process those messages. If you have multiple instances in your worker environment tier, each instance has its own daemon, but they all read from the same Amazon SQS queue.



# Deploying Elastic Beanstalk Applications from Docker Containers

Elastic Beanstalk supports the deployment of web applications from Docker containers. With Docker containers, you can define your own runtime environment. You can choose your own platform, programming language, and any application dependencies (such as package managers or tools), that aren't supported by other platforms. Docker containers are self-contained and include all the configuration information and software your web application requires to run.

By using Docker with Elastic Beanstalk, you have an infrastructure that automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

### Single Container Docker

Use the single container configuration when you only need to run one container per instance.

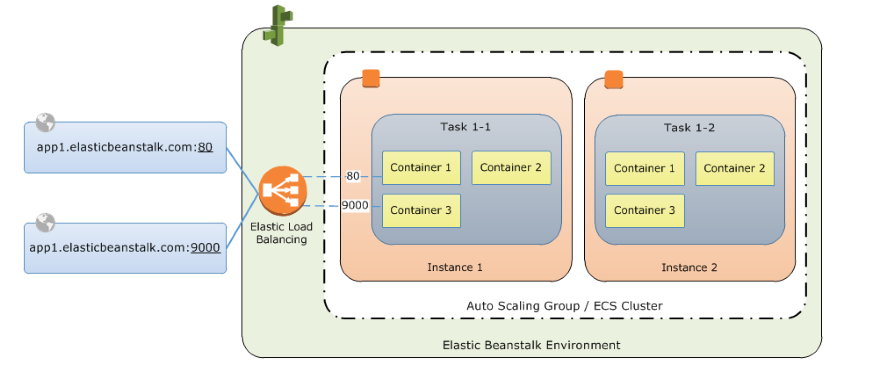
Single container Docker environments can be launched from a Dockerfile (which describes an image to build), a Dockerrun.aws.json file (which specifies an image to use and additional Elastic Beanstalk configuration options), or both. These configuration files can be bundled with source code and deployed in a ZIP file.

### Multicontainer Docker

Use the multicontainer configuration when you need to deploy multiple Docker containers to each instance.

Standard generic and preconfigured Docker platforms on Elastic Beanstalk support only a single Docker container per Elastic Beanstalk environment. In order to get the most out of Docker, Elastic Beanstalk lets you create an environment where your Amazon EC2 instances run multiple Docker containers side by side.

The following diagram shows an example Elastic Beanstalk environment configured with three Docker containers running on each Amazon EC2 instance in an Auto Scaling group:



## Configuring Additional Storage Volumes

For improved performance, Elastic Beanstalk configures two Amazon EBS storage volumes for your Docker environment's EC2 instances. In addition to the root volume provisioned for all Elastic Beanstalk environments, a second 12GB volume named xvdcz is provisioned for image storage on Docker environments.

If you need more storage space or increased IOPS for Docker images, you can customize the image storage volume by using the BlockDeviceMapping configuration option in the [aws:autoscaling:launchconfiguration](https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/command-options-general.html" \l "command-options-general-autoscalinglaunchconfiguration) namespace.