

◆ Get unlimited access to the best of Medium for less than \$1/week. [Become a member](#)



# AWS Lambda Tutorial — A Beginner's Guide To AWS Lambda



Vishal Padghan · [Follow](#)

Published in Edureka · 8 min read · Oct 28, 2016

28



AWS Lambda Tutorial - Edureka

Today we're going to talk about AWS Lambda. AWS Lambda is a compute service offered by Amazon. You must be curious as there are several other

compute services from AWS, such as *AWS EC2*, *AWS Elastic Beanstalk*, *AWS Opsworks* etc., then why another compute service? In this AWS Lambda tutorial you will discover what is AWS Lambda, why it is used and in which use cases you should consider it.

Let's see how Amazon defines AWS Lambda and then we will take a deep dive into the key concepts, understand a use case with a hands-on in the end.

## What is AWS Lambda?

Amazon explains AWS Lambda ( $\lambda$ ) as a 'serverless' compute service, meaning the developers, don't have to worry about which AWS resources to launch, or how will they manage them, they just put the code on lambda and it runs, it's that simple! It helps you to focus on core-competency i.e. App Building or the code.

## Where will I use AWS Lambda?

AWS Lambda executes your backend code, by automatically managing the AWS resources. When we say 'manage', it includes launching or terminating instances, health checkups, auto-scaling, updating or patching new updates etc.

## So, how does it work?

The code that you want Lambda to run is known as a **Lambda function**. Now, as we know a function runs only when it is called, right? Here, **Event Source** is the entity which triggers a Lambda Function, and then the task is executed.

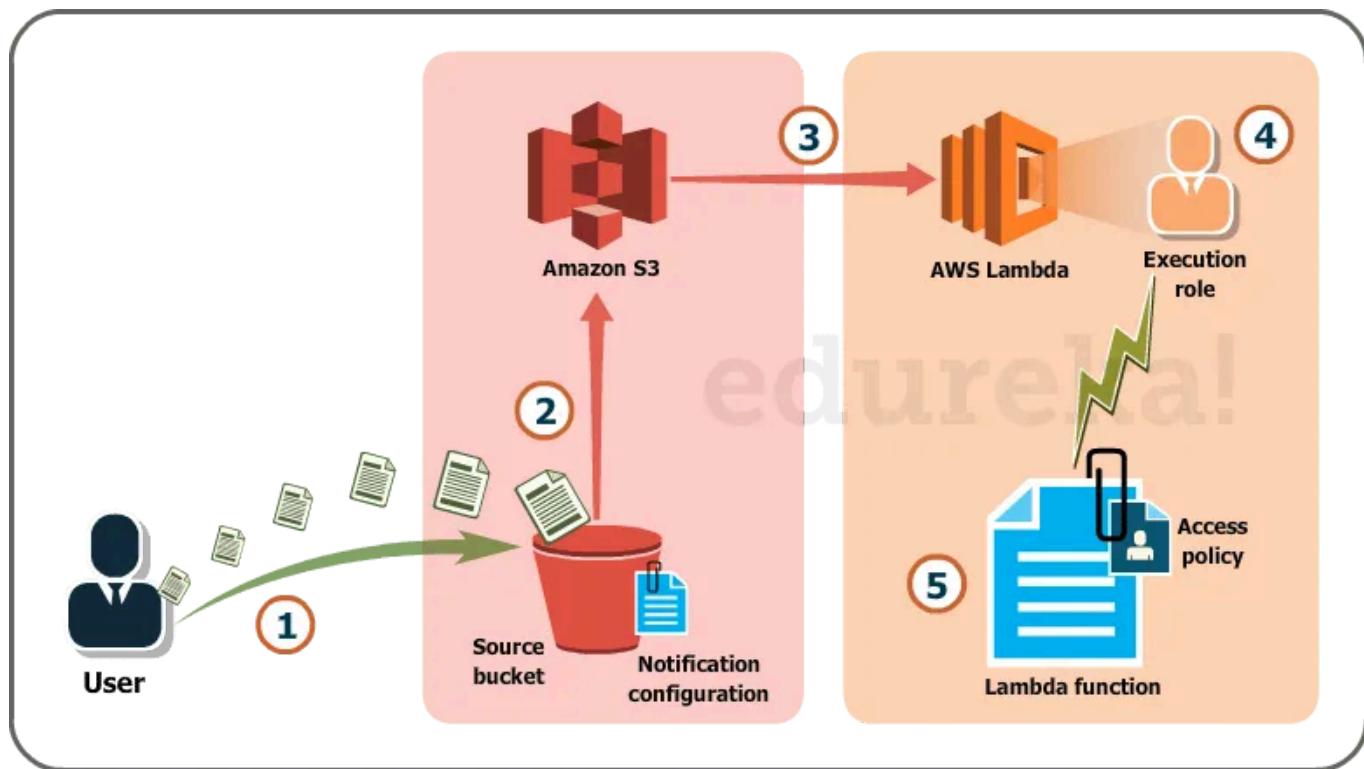
Let's take an example to understand it more clearly.

Suppose you have an app for image uploading. Now when you upload an image, there are a lot of tasks involved before storing it, such as resizing, applying filters, compression etc.

So, this task of uploading of an image can be defined as an **Event Source** or the ‘trigger’ that will call the Lambda Function, and then all these tasks can be executed via the Lambda function.

In this example, a developer just has to define the event source and upload the code.

Let's understand this example with real AWS resources now.



Over here we will be uploading images in the form of objects to an S3 bucket. This uploading an image to the S3 bucket will become an event source or the ‘trigger’.

The whole process, as you can see in the diagram, is divided into 5 steps, let's understand each one of them.

1. User uploads an image (object) to a source bucket in S3 which has notification attached to it, for Lambda.
2. The notification is read by S3 and it decides where to send that notification.
3. S3 sends the notification to Lambda, this notification acts as an invoke call of the lambda function.
4. Execution role in Lambda can be defined by using IAM (Identity and Access Management) to give access permission for the AWS resources, for this example here it would be S3.
5. Finally, it invokes the desired lambda function which works on the object which has been uploaded to the S3 bucket.

*If you were to solve this scenario traditionally, along with development, you would have hired people for managing the following tasks:*

- Size, provision and scale up group of servers
- Managing OS updates
- Apply security patches and
- Monitor all this infrastructure for performance and availability.

This would have been an expensive, tedious and tiresome task, therefore the need for AWS Lambda is justified. AWS Lambda is compatible with Node.JS, Python and Java, so you can upload your file in a zip, define an event source and you are set!

*We now know - How Lambda works and What Lambda does.*

Now, let's understand-

- Where to use Lambda?
- What purpose does Lambda serve, that other AWS Compute services don't?

If you were to architect a solution to a problem, you should be able to identify where to use Lambda, right?

So, as an architect you have the following options to execute a task:

- AWS EC2
- AWS Elastic Beanstalk
- AWS OpsWorks
- AWS Lambda

Let's take the above use case as an example and understand why we chose Lambda to solve it.

AWS OpsWorks and AWS ElasticBeanstalk are used to deploy an app, so our use case is **not** to create an app but to execute a back-end code.

### **Then why not EC2?**

If you were to use EC2, you would have to architect everything i.e. load balancer, EBS volumes, software stacks etc. In lambda you don't have to worry about anything, just insert your code, and AWS will manage the rest!

*For example,* in EC2 you would be installing the software packages on your virtual machine which would support your code, but in Lambda you don't have to worry about any VM, just insert plain code and Lambda will execute it for you.

But, if your code will be running for hours, and you expect a continuous stream of requests, you should probably go with EC2, because the architecture of Lambda is for a sporadic kind of workload, wherein there will be some quiet hours and some spikes in the no. of requests as well.

*For example,* logging the email activity for say a small company, would see more activity during the day than in the night, also there could be days when there are less emails to be processed, and sometimes the whole world could start emailing you! In both the cases, Lambda is at your service.

Considering this use case for a big social networking company, where the emails are never-ending because it has a huge user base, Lambda may not be the apt choice.

## **Limitations of AWS Lambda**

Some limitations are hardware specific and some are bound by the architecture, let's discuss all of them.

*Hardware limitations* include the disk size, which is limited to 512 MB, the memory can vary between 128 MB and 1536 MB. Then there are some other such as the execution timeout can be maximized to just 5 minutes, your request body payload can be not more than 6 MB and your request body is 128 KB. The request body payload is like the data that you send with a "GET"

or “PUT” request in HTTP, whereas the request body would be the type of request, the headers etc.

Actually, these are not limitations but are the design boundaries which have been set in the architecture of Lambda so if your use case does not fit these, you always have the other AWS compute services at your disposal.

We discussed in this AWS Lambda Tutorial that how doing tasks in Lambda are “not” tedious and tiresome. Let’s now cover the expense part as well.

## Pricing in AWS Lambda

Like most of the AWS services, AWS Lambda is also a pay per use service, meaning you only pay what you use, therefore you are charged on the following parameters

- The number of **requests** that you make to your lambda function
- The **duration** for which your code executes.

### Requests

1. You are charged for the number of requests that you make across all your lambda functions.
  2. AWS Lambda counts a request each time it starts executing in response to an event source or invoke call, including test is invoked from the console.
- Let’s look at the prices now:

- First 1 million requests, every month are for free.

- 0.20\$ per million requests thereafter.

## Duration

1. Duration is calculated from the moment your code starts executing till the moment it returns or terminates, it is rounded up to the nearest 100ms.
2. The price depends on the amount of memory you allocate to your function, you are charged \$0.00001667 for every GB-second used.

\* Source: AWS official website

If you have reached till here you are all set for a Hands-on in Lambda. Let's have some fun!

## Hands-on: AWS Lambda DIY

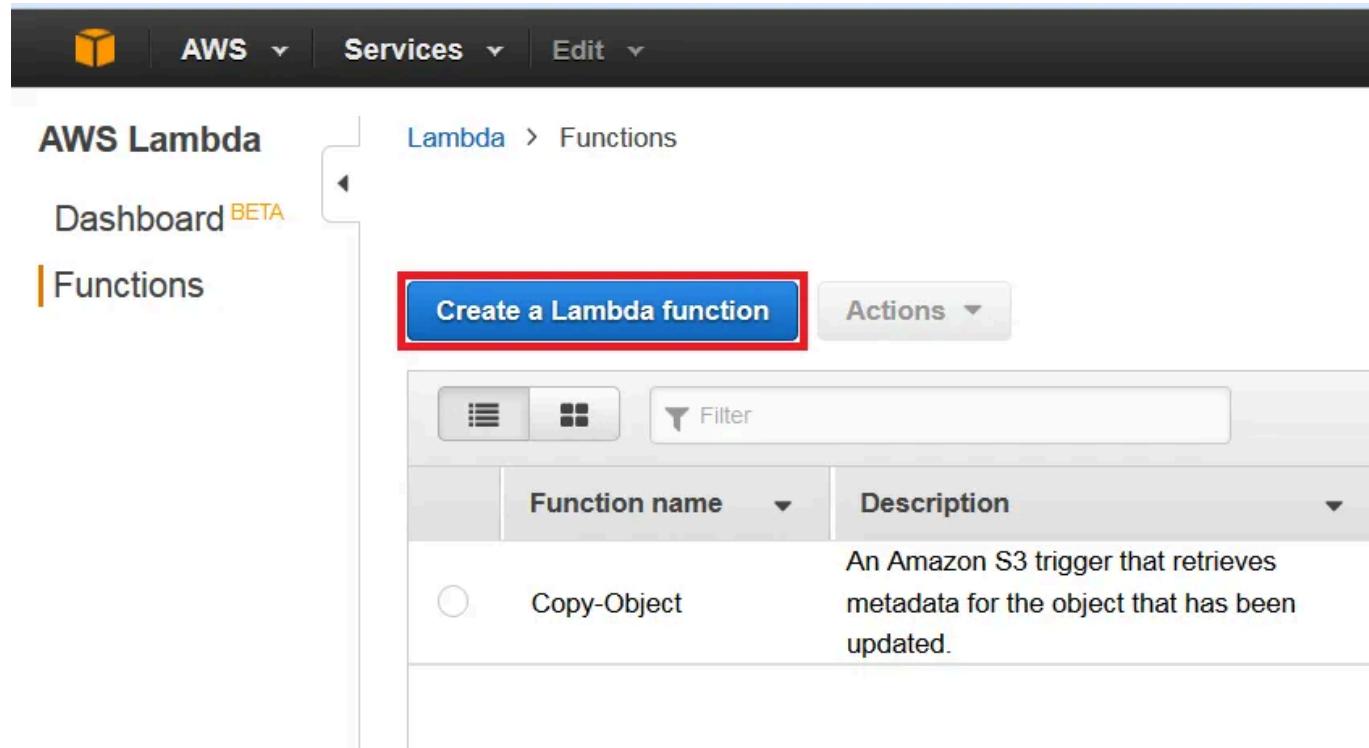
Let's create a Lambda function which will log "An object has been added" once you add an object to a specific bucket in S3.

**Step1:** From the AWS Management Console under compute section, select **AWS Lambda**.

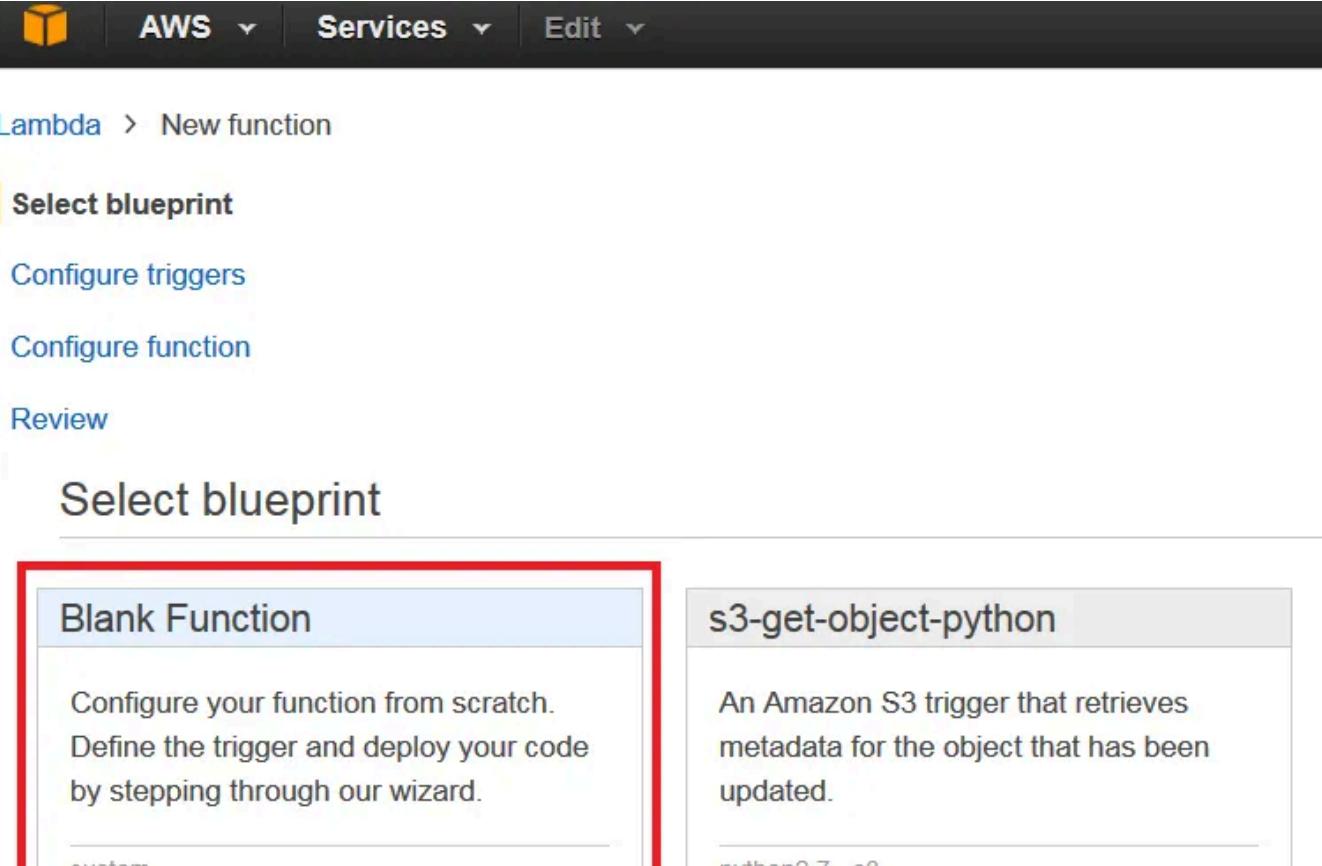
# Amazon Web Services

Compute	Developer Tools	Internet of Things
<b>EC2</b> Virtual Servers in the Cloud	<b>CodeCommit</b> Store Code in Private Git Repositories	<b>AWS IoT</b> Connect Devices to the Cloud
<b>EC2 Container Service</b> Run and Manage Docker Containers	<b>CodeDeploy</b> Automate Code Deployments	<b>Game Development</b>
<b>Elastic Beanstalk</b> Run and Manage Web Apps	<b>CodePipeline</b> Release Software using Continuous Delivery	<b>GameLift</b> Deploy and Scale Session-based Multiplayer Games
<b>Lambda</b> Run Code without Thinking about Servers	<b>Management Tools</b>	<b>Mobile Services</b>
<b>Server Migration</b> Migrate on-premises servers to AWS	<b>CloudWatch</b> Monitor Resources and Applications	<b>Mobile Hub</b> Build, Test, and Monitor Mobile Apps
Storage & Content	<b>CloudFormation</b> Create and Manage Resources with Templates	<b>Cognito</b> User Identity and App Data Synchronization

**Step2:** On the AWS Lambda Console, click on “Create a Lambda function”.

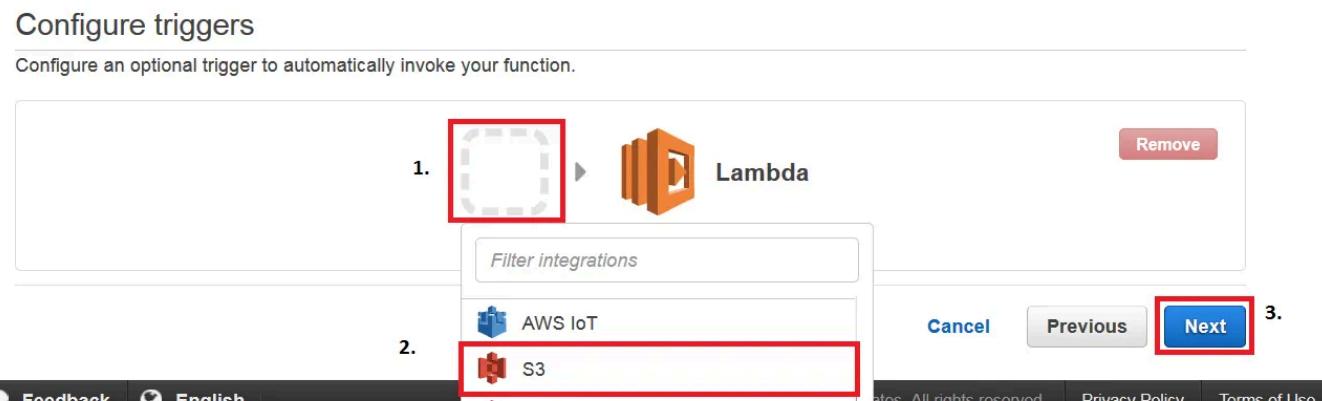


**Step3:** On the next page, you have to select a blueprint. For example, we will be selecting the blank function for our use-case.



The screenshot shows the AWS Lambda 'New function' wizard. At the top, there are navigation links: AWS, Services, and Edit. Below that, it says 'Lambda > New function'. The main section is titled 'Select blueprint' with four options: 'Configure triggers', 'Configure function', 'Review', and 'Select blueprint' (which is bolded). Under 'Select blueprint', there are two cards: 'Blank Function' (highlighted with a red box) and 's3-get-object-python'. The 'Blank Function' card contains the text: 'Configure your function from scratch. Define the trigger and deploy your code by stepping through our wizard.' Below it is a 'custom' link. The 's3-get-object-python' card contains the text: 'An Amazon S3 trigger that retrieves metadata for the object that has been updated.' Below it is a 'python2.7 · s3' link.

**Step4:** On the next page you will be (1) setting a trigger, since we are going to work on S3, (2) select the S3 trigger and then (3) click Next.



The screenshot shows the 'Configure triggers' step in the AWS Lambda wizard. It has three numbered steps: 1. A placeholder icon with a dashed border, followed by a Lambda icon and the word 'Lambda'. 2. A list of triggers: AWS IoT, S3 (which is highlighted with a red box), and SNS. 3. A 'Next' button at the bottom right, which is also highlighted with a red box. Below the steps, there are links for 'Feedback', 'English', and other footer information.

**Step5:** On the configuration page, fill in the details. You can put your own code, or you can copy the same code from this use-case. After that, fill the handler and role, leave the advanced settings as it is, in the end click next.

## Configure function

A Lambda function consists of the custom code you want to execute. [Learn more about Lambda functions.](#)

Name*	ex-lambda
Description	A function to log the addition of an object in
Runtime*	Node.js 4.3

## Lambda function code

Code entry type	Edit code inline
<pre>1 exports.handler = (event, context, callback) =&gt; { 2   console.log('An object in S3 is added'); 3 };</pre>	

## Lambda function handler and role

Handler*	index.handler
Role*	Create new role from template(s)
Role name*	ex-lambda
Policy templates	S3 object read-only per...

\* These fields are required.

[Cancel](#)

[Previous](#)

[Next](#)

**Step6:** On the next page, review all the information, and click on “Create function”.

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

**Step7:** Now, since we created the function for S3 bucket, the moment you add a file to your S3 bucket, you should get a log for the same in CloudWatch, which is a monitoring service from AWS.

Filter events		<a href="#">all</a> 30s 5m 1h 6h 1d 1w custom ▾
Time (UTC +00:00)	Message	
2016-10-25	<i>No older events found at the moment. <a href="#">Retry</a>.</i>	
▶ 15:18:08	START RequestId: 3bdd7540-9ac6-11e6-ba06-f9f5aabe9def Version: \$LATEST	
▼ 15:18:08	2016-10-25T15:18:08.416Z 3bdd7540-9ac6-11e6-ba06-f9f5aabe9def An object in S3 is added	
2016-10-25T15:18:08.416Z 3bdd7540-9ac6-11e6-ba06-f9f5aabe9def	An object in S3 is added	
▶ 15:18:08	END RequestId: 3bdd7540-9ac6-11e6-ba06-f9f5aabe9def	
▶ 15:18:08	REPORT RequestId: 3bdd7540-9ac6-11e6-ba06-f9f5aabe9def Duration: 17.23 ms Billed Duration: 100 ms Memory Size: 128 M	
<i>No newer events found at the moment. <a href="#">Retry</a>.</i>		

**Congratulations! You have successfully executed the Lambda Function.**

I hope you enjoyed the deep dive into the AWS Lambda Tutorial. It is one of the most desired knowledge areas in AWS ecosystem for job positions such as Solutions Architect, Cloud Engineer, DevOps Engineer.

If you wish to check out more articles on the market's most trending technologies such as Artificial Intelligence, DevOps, Ethical Hacking, then you can refer to Edureka's official site.

Do look out for other articles in this series which will explain the various other aspects of AWS.

## 1. [AWS Tutorial](#)

2. [AWS EC2](#)

3. [AWS ResUME](#)

4. [AWS Elastic Beanstalk](#)

5. [AWS S3](#)

6. [AWS Console](#)

7. [AWS RDS](#)

8. [AWS Migration](#)

9. [AWS Fargate](#)

10. [Amazon Lex](#)

11. [Amazon Lightsail](#)

12. [AWS Pricing](#)

13. [Amazon Athena](#)

14. [AWS CLI](#)

15. [Amazon VPC Tutorial](#)

15. [AWS vs Azure](#)

17. [On-premise vs Cloud computing](#)

18. [Amazon Dynamo DB Tutorial](#)

19. [How To Restore EC2 From Snapshot?](#)

20. [AWS CodeCommit](#)

21. [Top AWS Architect Interview Questions](#)

22. [How To Restore EC2 From Snapshot?](#)

23. [Create Websites using AWS](#)

24. [Amazon Route 53](#)

25. [Securing Web Applications With AWS WAF](#)

*Originally published at [www.edureka.co](http://www.edureka.co) on October 28, 2016.*

AWS

Amazon Web Services

Aws S3

Aws Ec2

AWS Lambda



## Published in Edureka

[Follow](#)

5.3K Followers · Last published Jul 30, 2022

There are many e-learning platforms on the internet & then there's us. We are not the biggest, but we are the fastest growing. We have the highest course completion rate in the industry. We provide live, instructor-led online programs in trending tech with 24x7 lifetime support.



## Written by Vishal Padghan

[Follow](#)

351 Followers · 1 Following

A keen writer and a Cloud Computing Enthusiast

## No responses yet



What are your thoughts?

[Respond](#)

## More from Vishal Padghan and Edureka



In Edureka by Vishal Padghan

## AWS Fargate—A Beginner's Guide To AWS Elastic Container Service

This blog on AWS Fargate will cover the need and working of AWS Fargate with a hands-on.

Nov 27, 2018

54



In Edureka by Shubham Sinha

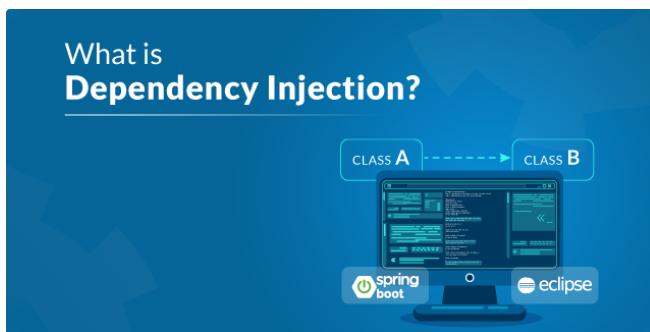
## Fundamentals of MapReduce with MapReduce Example

In this MapReduce Tutorial you will learn all about MapReduce such as what is...

Nov 15, 2016

381

3



In Edureka by Swatee Chand

## Dependency Injection Using Spring Boot

This article on Dependency Injection is a comprehensive guide to the technique with ...

Jul 2, 2019

407

6



In Edureka by Vishal Padghan

## AWS Elastic Beanstalk—Deploying An Application Using Beanstalk

This blog on AWS Elastic Beanstalk will tell you how to deploy applications without...

Oct 23, 2018

43

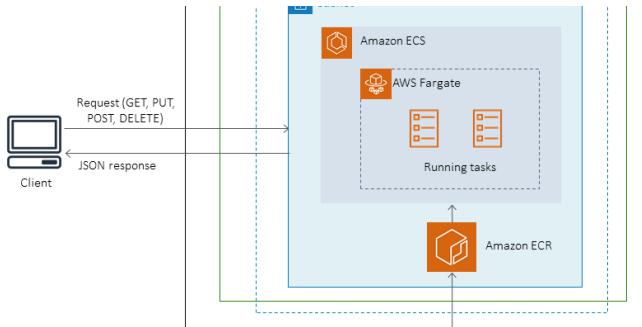
2



See all from Vishal Padghan

See all from Edureka

## Recommended from Medium



### AWS S3: The Foundation of Object Storage



- S3: A Closer Look
- S3 Consistency Model
- Choosing the Right Storage Class
- Data Availability and Durability
- S3 Versioning: Protecting Your Data
- Securing Data in S3
- Optimizing S3 for Cost and Performance

 Algorithm Alchemist

## Building Microservices and a CI/CD Pipeline with AWS

Modernizing applications by transitioning from monolithic architectures to...

Jan 25  5

 In AWS in Plain English by Pritam Deb

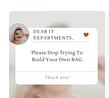
## AWS S3 Interview Guide: Key Concepts and Practical Insights f...

Understanding AWS S3: Storage Classes, Data Security, and Cost Optimization Tips fo...

 Aug 25, 2024  2

## Lists



### Natural Language Processing

1912 stories • 1569 saves

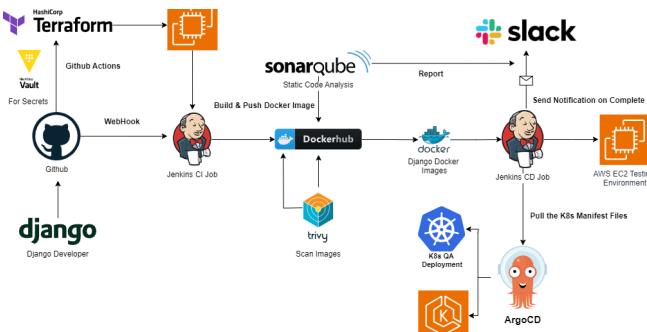


 Rohit Shrivastava

## #38 Shorticles: Application Configuration Security in AWS

In AWS, application configuration typically includes the following elements:

Sep 23, 2024



 In Django Unleashed by Joel Wembo

## Technical Guide: End-to-End CI/CD DevOps with Jenkins, Docker,...

Building an end-to-end CI/CD pipeline for Django applications using Jenkins, Docker,...

 Apr 12, 2024  1K  21

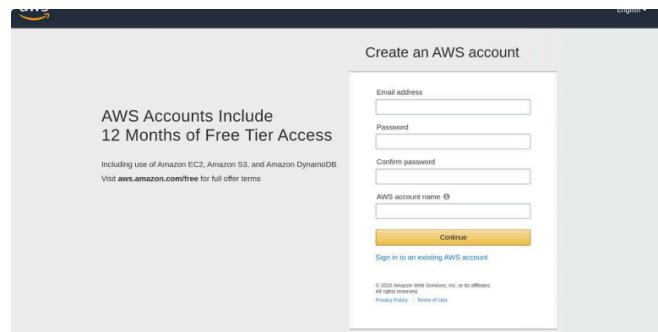


 FinanceAndCode

## AI Engineer Career Roadmap: A Comprehensive Guide

The field of Artificial Intelligence (AI) is rapidly evolving, offering myriad opportunities for...

 Sep 22, 2024



 M. Esat Ceber

## System Design with Amazon Web Services—2. AWS Basics

Amazon Web Services (AWS) is a cloud-based service. With this service, you can store data,...

 Aug 21, 2024



[See more recommendations](#)