

Learn AWS Amplify

The Basics



AWS Amplify is a command-line utility that generates web and mobile applications on Amazon Web Services (AWS) servers. It is written in node and requires NodeJS. My recommendation is to [use nvm to install NodeJS](#) if you're installing NodeJS for the first time.

Use Cases

- You posted your app idea on Craigslist, but no developers are willing to work for free, so you'll do it yourself!
- You want to create a proof of concept application quickly.
- You would like to evaluate several different technologies without having to learn the internals.
- You want to do something more than basic form collection.

You might want to [evaluate Honeycode](#) if you're doing basic data collection from web and mobile platforms.

Installation

The latest and greatest [installation instructions for a react](#) application work well from beginning to end. The last time I tried to learn AWS Amplify and follow the AWS Amplify instructions, it ended in a whole lot of errors. This time around, everything worked as expected.

Amplify Choices

When creating an amplify application you can build either a web or a mobile application. In the case of the mobile applications you can base your apps on web frameworks like React and Ionic but also gain the ability to manipulate system level APIs.

Amplify Console

Running this command pops you into the AWS console.

```
amplify console
```

I've found that clicking on the app name after running this command is the most useful route. I'm able to get to the build status of the project since I decided to automated my deployment with a GitHub repository.

Push

If you're looking to update your API you'll want to push up the changes with this command.

```
amplify push
```

Update

If you've made changes to your schema you'll want to update your API.

Run this command to push the changes and generate your schema.

```
amplify update api
```

Wrapping Up

I wrote a short article because there isn't a lot to explain with AWS Amplify at a high level. It can get complicated when you're working to keep your resources in order. If you're not looking to do something complicated and want to do some data collection with a few bells and whistles, AWS Amplify will fit the bill.

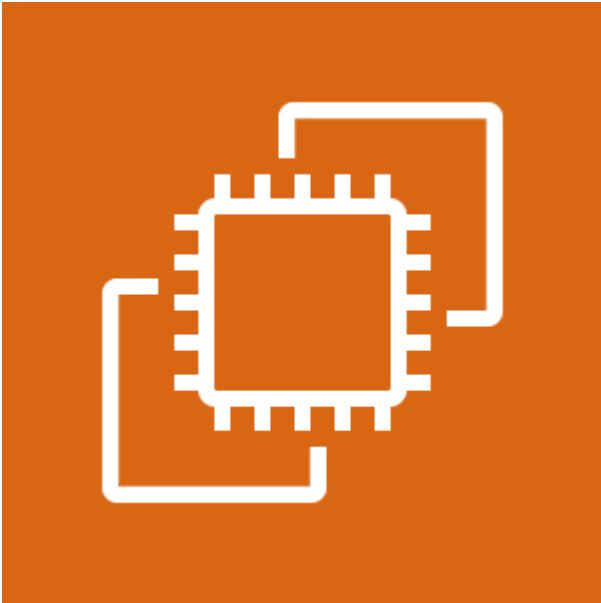
Learn AWS Amplify – Beyond the Basics

So far there aren't many resources for learning AWS Amplify but here are a few.

- [AWS Amplify: Console User Guide](#)
 - [Learning AWS Amplify](#) (LinkedIn)
 - [Serverless React with AWS Amplify – The Complete Guide](#) (Udemy)
 - [AWS AppSync & Amplify with React & GraphQL – Complete Guide](#) (Udemy)
-

Learn EC2

The Basics



Learn EC2 basics, concepts, and more.

Amazon EC2 is the core compute service provided by Amazon Web Services. It's [Virtual Machines](#) in the cloud.

If you have been using Amazon Web Services for a while there is no doubt you encountered EC2 in one way or another. It is a central service used by [Amazon Lightsail](#) and numerous other offerings.

Primary Use Cases

1. Hosting environments for your apps
2. High-performance computing
3. Computing with more specific use cases like:
 1. GPU Heavy Tasks
 2. High Memory Requirements

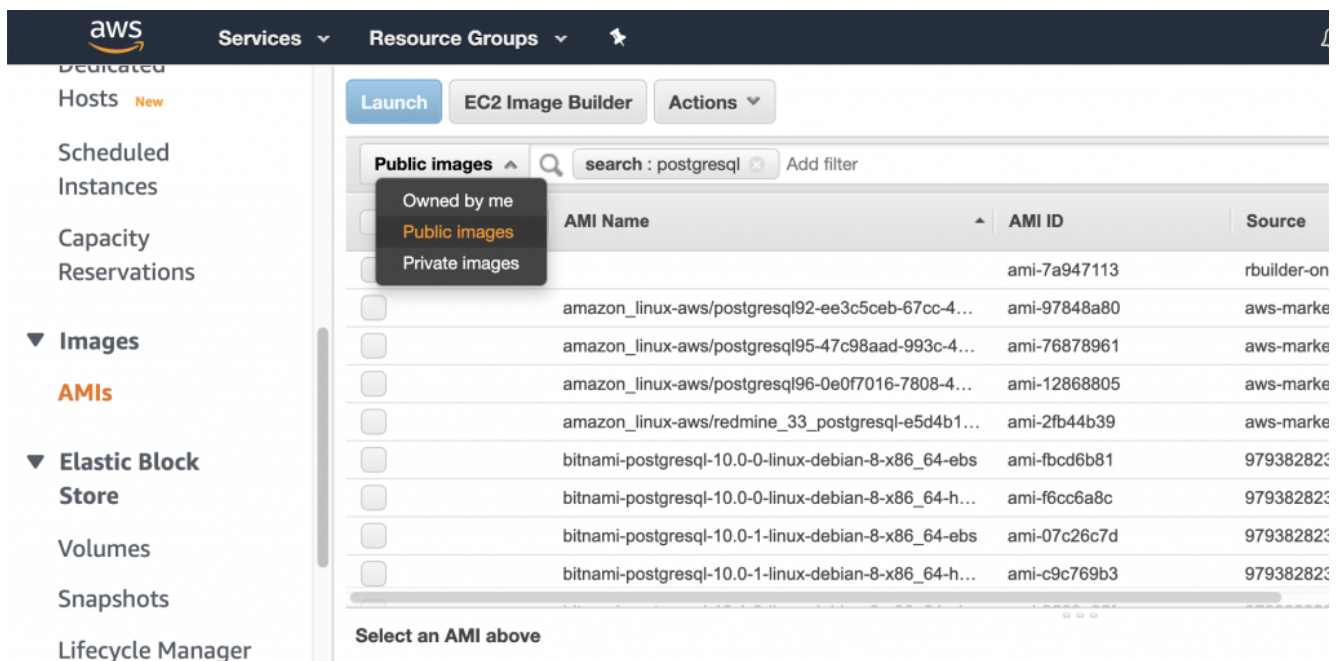
3. Burstable compute

Instances

EC2 Instances are central to the ecosystem. To use EC2, you will launch an EC2 instance from an EC2 image. An EC2 image contains a snapshot of an operating system like Linux with everything configured as it was upon snapshot creation.

Images

Amazon Machine Images (AMI) are prepackaged in numerous ways to meet a legion of use cases. Public images are found within [the AWS console's AMI search feature](#). If you have an AWS account, you can access the search feature.



Security Groups

Security groups are a collection of [firewall](#) rules that secure your instances from nefarious hackers. If you're familiar with firewalls, then you will be familiar with security groups. Apply [the principle of least privilege](#) and only grant access to systems that require access to your EC2 instances.

Other Concepts

Learn EC2 concepts you'll need to understand when using EC2.

Key Pairs

Use key pairs to gain [secure shell](#) access on your EC2 instance.

Volumes

Attach disks to your EC2 instances to expand space and meet various use cases like high performance read/write disk access.

Load Balancers

Scale up your instances by routing traffic to multiple instances.

Auto Scaling Groups

A definition of how many instances you want to run within a cluster along with thresholds on when to expand to the size of the cluster.

Snapshots

Create a moment in time snapshot of your EC2 instances.

Capacity Reservations

Save money by reserving your EC2 instances in advance.

Learn EC2 – Beyond the Basics

- **Books**

- [Programming Amazon Ec2](#) (O'Reilly)

- **Videos**

- [Amazon EC2 Fundamentals](#) (LinkedIn)
- [Managing AWS EC2 Instances](#) (Pluralsight)

Related Content

Cloud providers

The Basics



Cloud providers have taken the tech world by storm over the past several years. Amazon led the way with cloud computing in 2006 when it launched [the Amazon Elastic Compute Cloud \(EC2\) service](#). The cloud, however, [was mentioned as early as 1998](#).

[The Flexera state of the cloud report](#) provides a snapshot of the current state of cloud computing. The [Flexera](#) report is compiled once a year and contains insights about how companies are using cloud providers, along with adoption rates.

Primary Use Cases

Companies use cloud providers for numerous reasons. There are many use cases due to the flexibility and interoperability cloud services provide.

Here are some of the primary use cases.

- Hosting servers on virtual machines and containers
- Solving large scale computing problems like big data analysis, machine learning, and artificial intelligence.

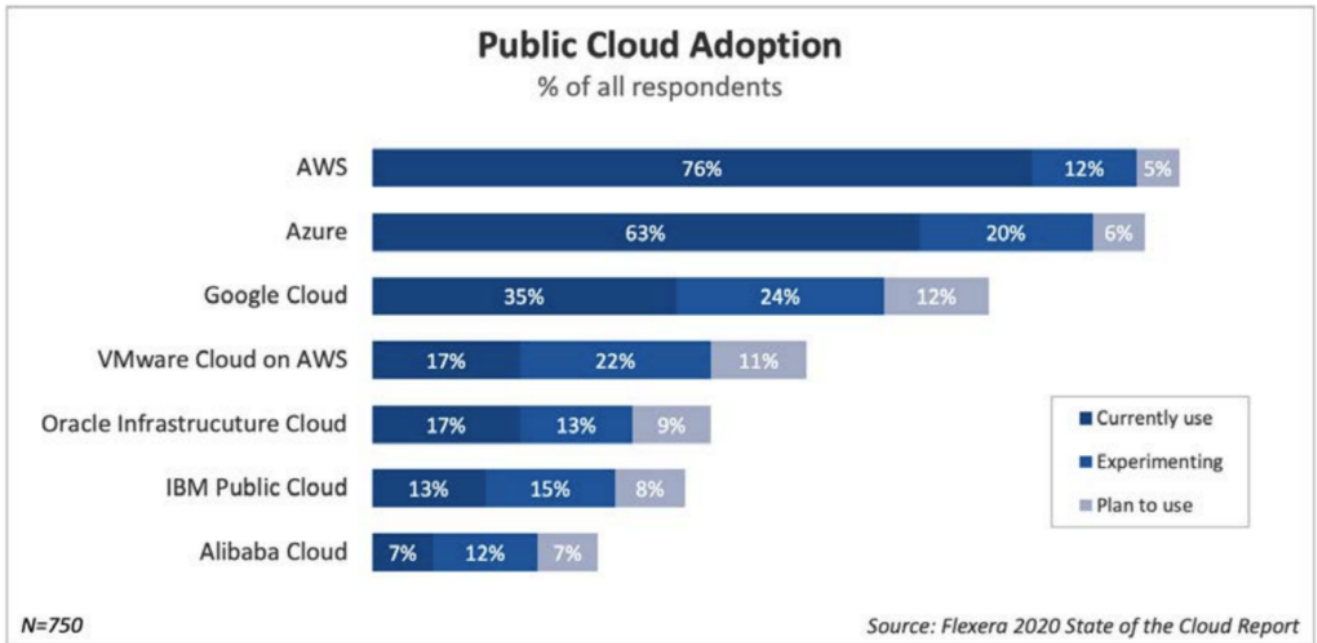
Adoption Rates

Cloud providers have enjoyed massive growth due to rapid adoption amongst companies going through digital transformations. Cloud adoption is mixed between public and private clouds.

Public Cloud Adoption

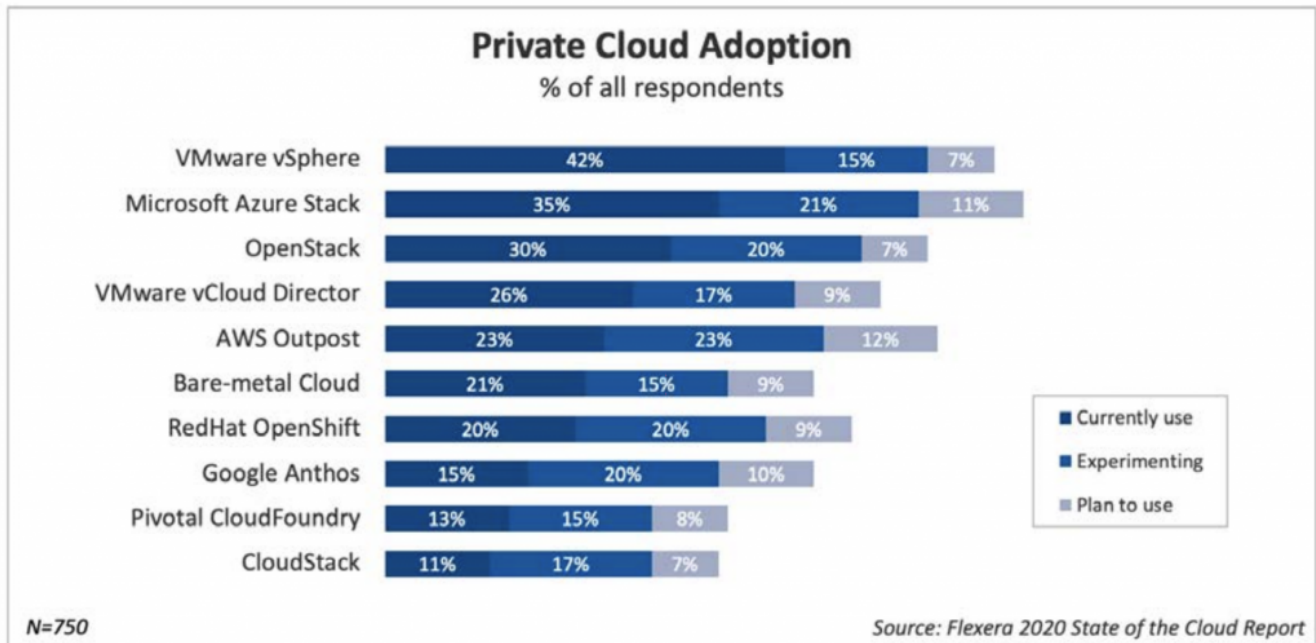
Amazon Web Services leads the pack while Microsoft Azure is

not far behind. Google has a fair share but lags due to a late start and a reduced number of relationships with large enterprises.



Private Cloud Adoption

Private clouds are an alternative to public clouds. These services allow people to reduce the complexity of hosting varied resources into a single-entry point of APIs, much like AWS, Azure, and other public cloud providers.



Amazon Web Services



In 2002 the AWS platform was born. In my eyes, this move was a classic case of [dogfooding](#) where Amazon needed [web scale](#) to deliver their shopping experience at scale. Why wouldn't Amazon attempt to recoup some of their hosting costs by reselling their platform?

The AWS platform applies [the single-responsibility principle](#) to its services in most cases. Alternatively, services like [LightSail](#) aggregate services into a single service.

Microsoft Azure



Microsoft Azure is like AWS in many ways. It provides scalable services that complement their business productivity solutions. Azure's cloud services tend to appear like a monolith within the [Azure Web Portal](#). Under the hood, there are single responsibility services like Azure Functions that allow you to run functions in the cloud, much like AWS with AWS Lambda.

Google Cloud



Google's services have evolved in fits over the years. [Google App Engine](#) was released in 2008 and was meant solely for Python applications with its first iteration. Fast forward to the present, and [Google App Engine](#) now supports a multitude of languages. Google's cloud services bear a resemblance to other cloud services, as evidenced by the [Google Cloud Functions service](#).

The Rest

I could [go on and on like Larry Ellison](#), but I'll spare you that. Luckily, the software industry is a shining bastion of commerce. There is plenty of room for the likes of Oracle, VMWare, Red hat, and others. The industry is booming, and competition is fierce just the way I like it.

Conclusion

Cloud providers are a continually evolving beast. Trends come and go, and the war rages on between cloud-native vs. containers. Developers are the clear winner of the cloud computing wars. If we can keep choice overload at bay, we will continue to benefit from the power these platforms provide.

Cloud Providers – Beyond the Basics

▪ Books

- [Digital Transformation: Survive and Thrive in an Era of Mass Extinction](#) (Amazon)
- [A Brief Guide to Cloud Computing](#) (Amazon)

▪ Videos

- [Fundamentals of Cloud Computing](#) (Pluralsight)
- [Cloud Computing: The Big Picture](#) (Pluralsight)
- [Learning Cloud Computing: Core Concepts](#) (LinkedIn)
- [Cloud Computing Concepts](#) (Udemy)

Related Content

Learn Amazon Athena

The Basics



Learn Amazon Athena basics. Amazon Athena is meant for querying copious amounts of data on the cheap. It's a straightforward service based on [Presto](#) with some interesting integrations. The Presto Foundation calls Amazon Athena and Amazon EMR [Presto Cloud on their website](#).

It's [not dirt cheap](#) to use Amazon Athena, but it is convenient. It costs \$5.00 per TB of data scanned. It's possible to pay a third of that with compression. Additional savings occur when querying a single column. The big win with Athena is the cluster of servers that AWS has at the ready. It wouldn't be straightforward to set up your own [Presto](#) analytics engine.

Primary Use Cases

- Querying vast amounts of
 - Log Data
 - Behavioral Data
 - Noncritical data

Importing Data

I'm facing a scenario where I need to query a 58G CSV file. I was able to load the ~1 billion rows into PostgreSQL, but any queries on the data caused my work machine to crash. I wrote a little python script to query the data which is working but is taking too long to process. So, I brought out the big guns.

I decided to try Athena out as a solution to my problem. Here's how it went down.

I created this table to store my CSV file. This table is for storing the output of an AWS command-line query.


```
DROP TABLE learnamazonathena.s3_objects;
CREATE EXTERNAL TABLE IF NOT EXISTS
learnamazonathena.s3_objects (
    time string,
    bytes bigint,
    object string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
STORED AS TEXTFILE
LOCATION 's3://learn-amazon-athena/csv';
```

Amazon S3 is now the datastore for our CSV file. We generate this CSV file with these commands.

```
# Query all S3 objects recursively
aws s3 ls --recursive learn-amazon-athena > s3_objects.csv

# Format the output as a CSV file
perl -p -i -e
's/(^.{0,19})(\s+)([0-9]+)(\s)(.+)$/"\1","\3","\5"/g'
applications.csv
```

I then ran this query against the newly created table.

```
SELECT split(object,
            '/') [2], SUM(bytes)
FROM learnamazonathena.s3_objects
GROUP BY split(object, '/') [2]
```

This query took seventeen seconds to complete.

Athena Query editor Saved queries **History** Data sources Workgroup : primary Settings Tutorial Help What's new

History

Search for name, query, etc.

Query submitted time	Query	Encryption type	State	Run time(s)	Data scanned	Action
2020/06/06 13:14:27 UTC-7	<code>SELECT split(object, '/') [2], SUM(bytes) from learnamazonathena.s3_objects GROUP By split(object, '...'</code>	N/A	Succeeded	17.41	58.08 GB	Download results



I love how fast and straightforward Amazon Athena is. The CSV file I created is small in the grand scheme of things. Facebook created Presto to query obscene amounts of data then later open-sourced it.

Learn Amazon Athena – Beyond the Basics

- **Print**

- [Getting Started – User Guide](#) (AWS)
- [Amazon Athena – User Guide](#) (AWS)
- [Athena FAQ](#) (AWS)

- **Video**

- [Amazon Web Services: Data Analytics](#) (LinkedIn)