How to Build a Serverless Website with AWS Lambda in 7 Easy Steps

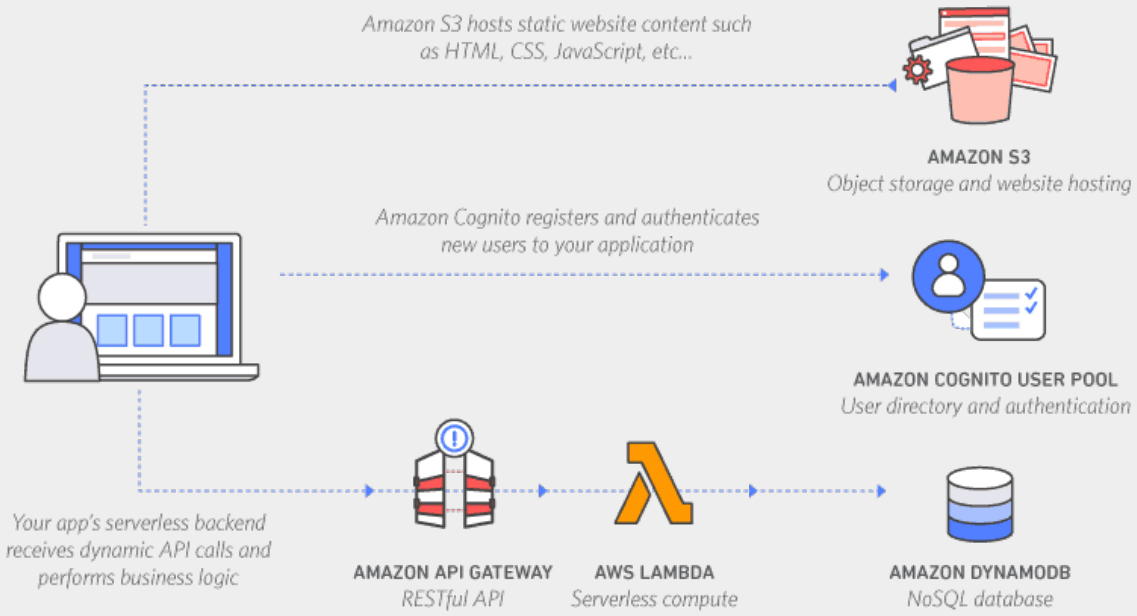
Serverless Computing allows you to concentrate on your application code instead of managing servers or installing any software. You can build a Serverless Website by using AWS Lambda, a serverless compute service and it’s simply great.

I’ll take you through the entire process of configuration and uploading your application code to AWS. So, without further ado, lets get started.

Build Your First Serverless Website in 7 Easy Steps

There are a few simple steps you will have to take in order to get up and running, so let’s walk the walk together and start with taking a look at the road ahead.

Here is an example of Serverless Application Architecture



Source: AWS Serverless Application Model

Prerequisites

- Aws Free Tier Account
- Install Nodejs v8+ and nmp

I'm using the serverless framework CLI to create and deploy apps.

Please make sure that npm is installed correctly and if you encounter any errors, check nmpjs documentation or execute the following commands to solve common nmp related problems quickly;

```
npm cache clean --force
```

```
npm i npm@latest -g
```

Serverless Setup

We have to install Serverless globally, so fire up a terminal window and run:

```
npm install -g serverless
```

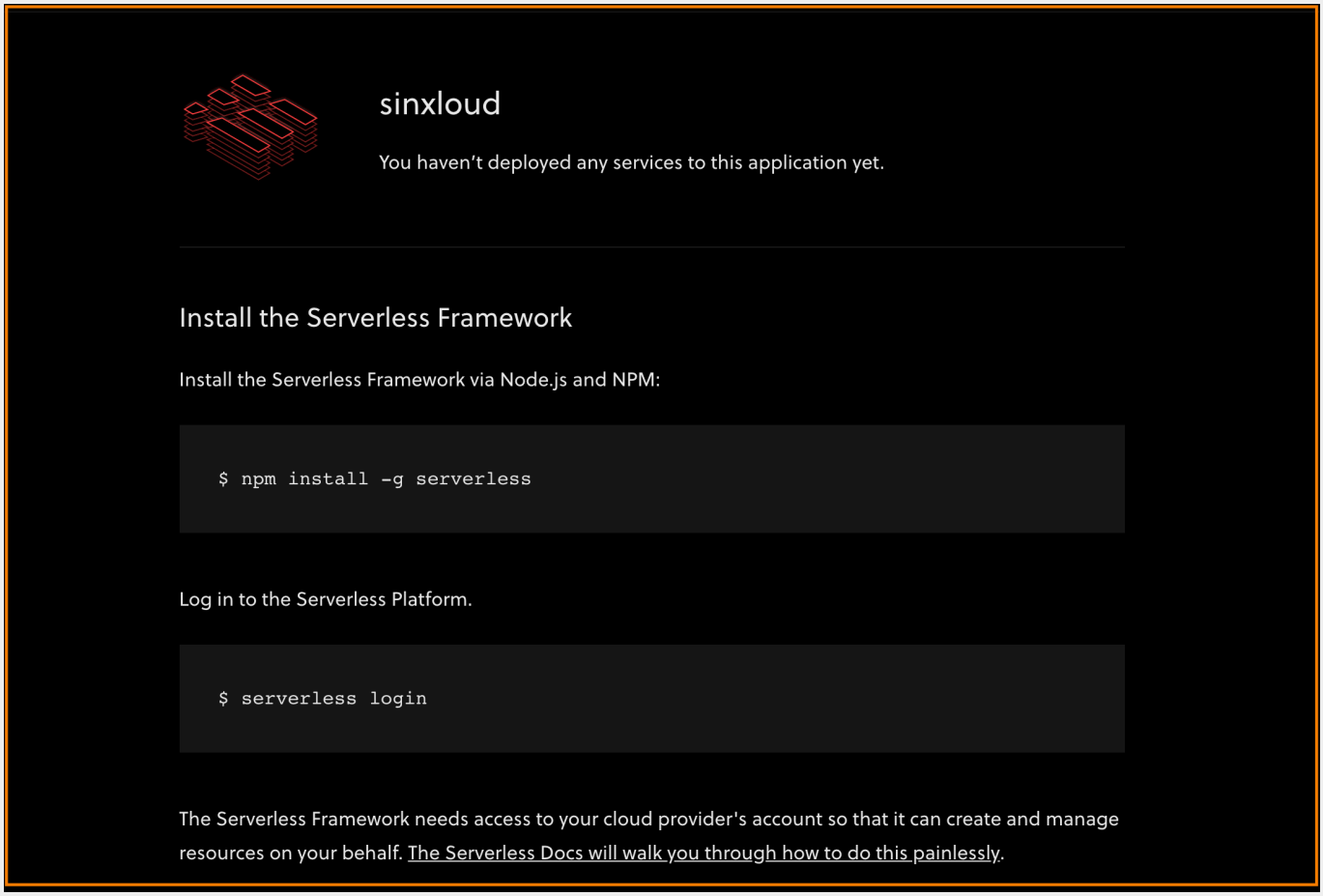
Note: You may need to run the command as sudo.

Next up we'll login into our newly installed serverless platform, type the following command:

```
serverless login
```

You'll have a new browser window open up asking you to log in (you can use GitHub or Google account for authentication).

You can follow on screen instructions to create a username and app name, once completed you will see default information page, similar to this:



Source: Serverless Platform

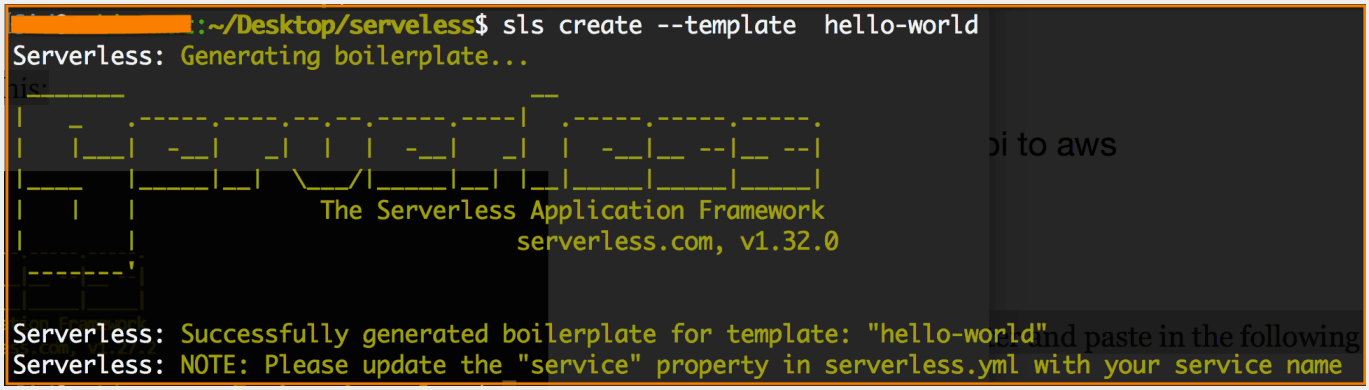
After that's done, you can create a new folder for the project and we need to create a simple serverless boilerplate.

```
mkdir serverless-sinx
```

```
cd servemeless-sinx
```

```
sls create --template hello-world
```

Once done, you'll end up with something similar to this:



Source: my mac :-)

Next we need to get our AWS credentials configured. Let's jump in and see it in action.

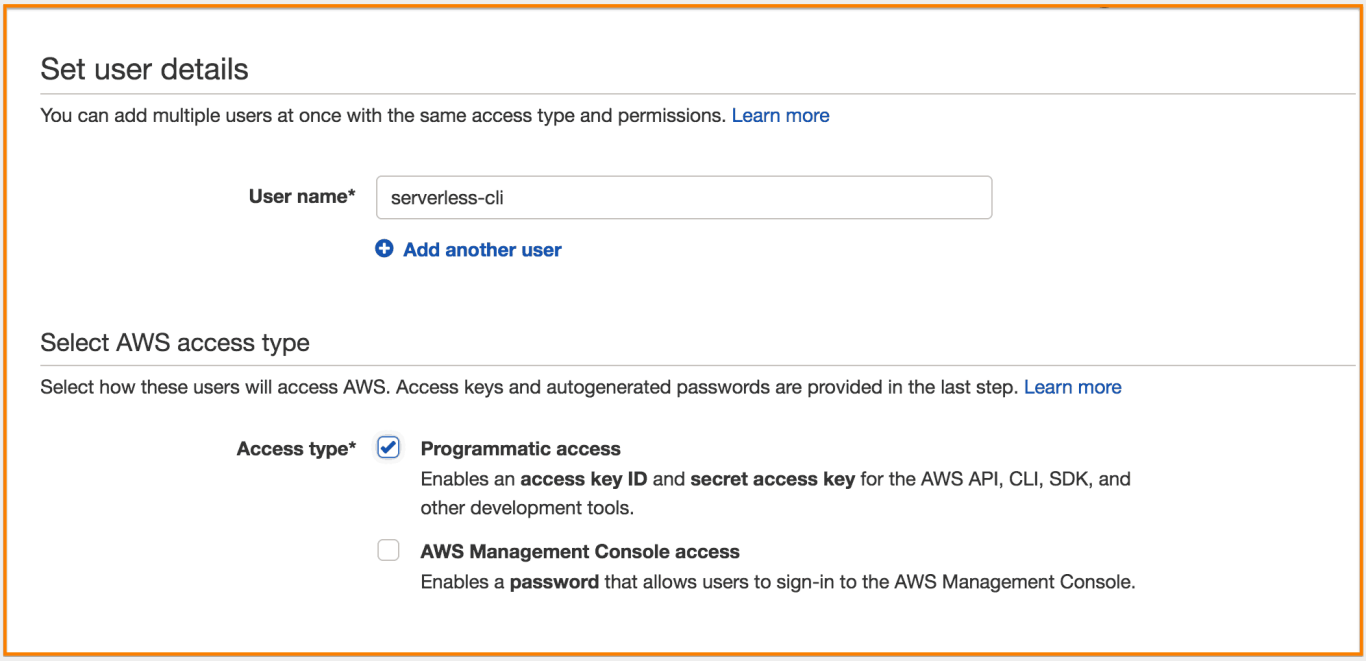
Configure AWS Credentials

The process is simple but requires multiple steps. Navigate to your AWS Console and click on Services dropdown in the top left corner.

You'll see a ton of services show up, Go ahead and select IAM under *"Security Identity and Complaince"* or write IAM in the search box and use the following instructions to create a user and grant necessary privileges..

Set User Details

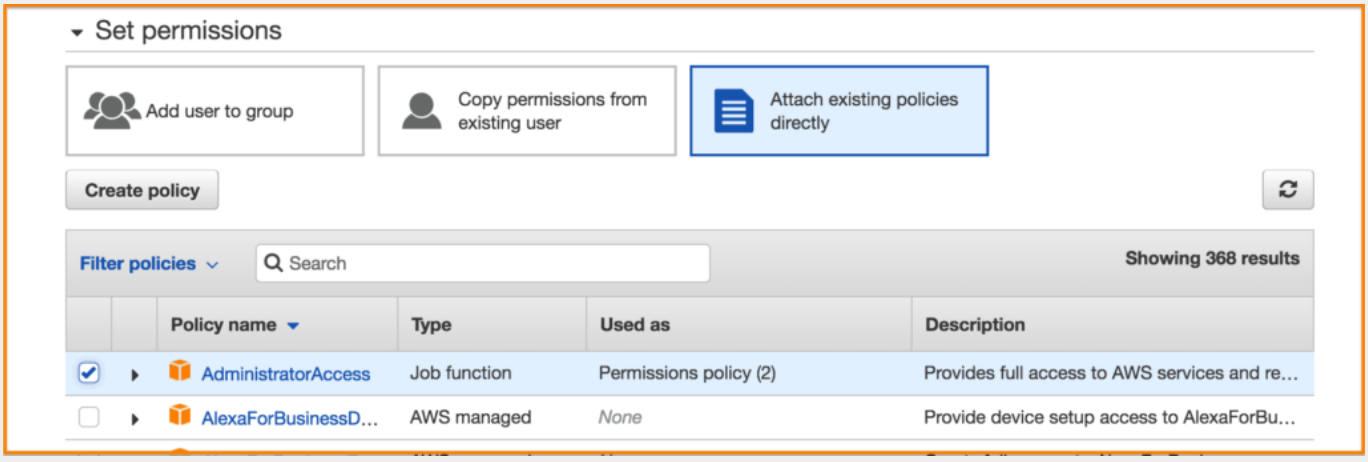
Create a new username and only select Programmatic access to continue.



Source: AWS Console

Set Permissions

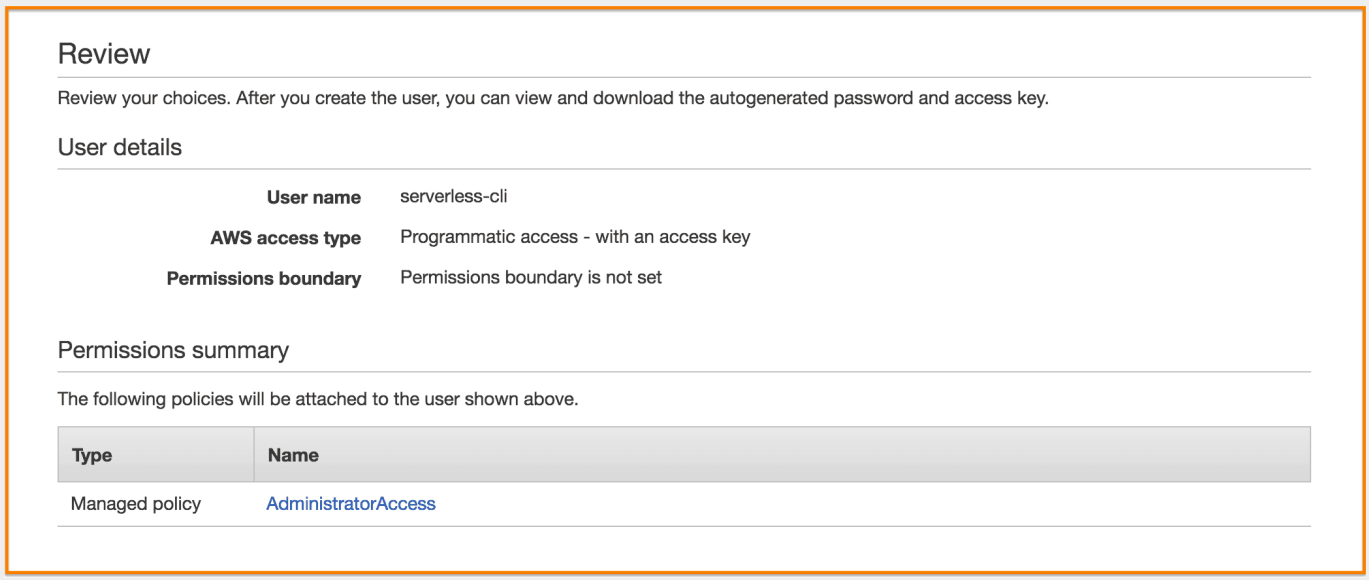
Go ahead and click on Attach existing policies directly and only select AdministratorAccess under the Policy section to continue.



Source: AWS Console

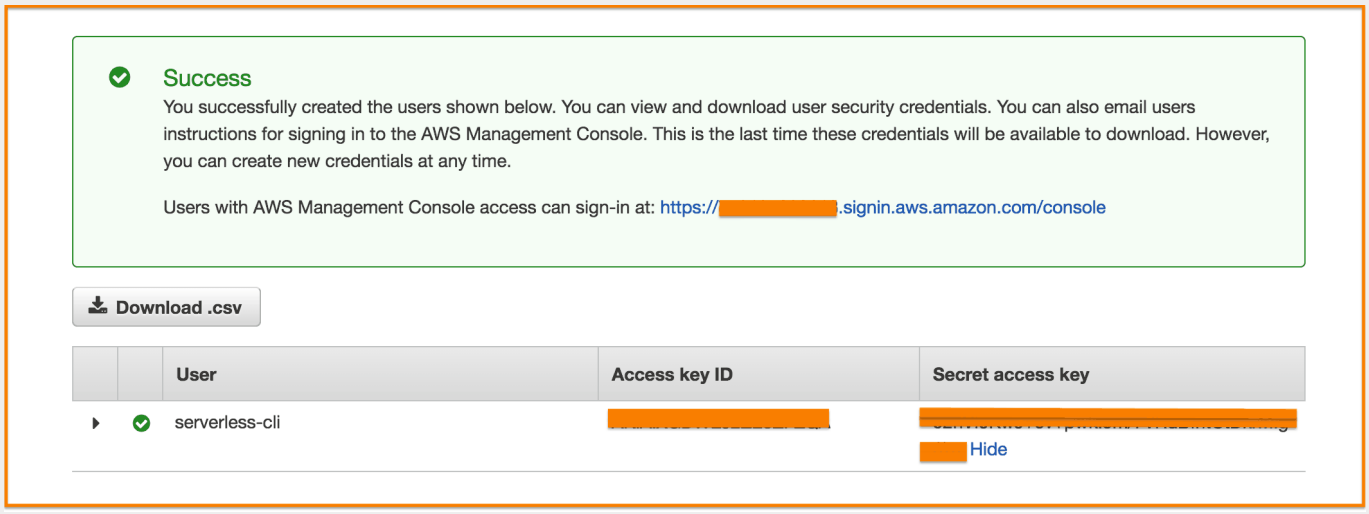
Complete Setup

Review the details to continue to user setup.



Source: AWS Console

Proceeding to the next step you will see the user was created. Now, and only now will you have access to the users Access Key ID and Secret Access Key.



Source: AWS Console

Save IAM keys in the Serverless configuration

Fire up terminal and type the following command in one line:

Now your Serverless installation is configured with AWS account.

Project Setup

Before we start installing dependencies for our serverless website, you will need to create a package.json file by executing the following command in terminal.

```
npm init
```

You'll be asked to provide names and descriptions and a lot of other information. Since this is a test just hit "Enter" and leave all the fields empty.

Installing Dependencies

We are going to install express web framework to get things going faster, install some dependancies and edit some files. Type the following in your terminal:

Install Express Framework

```
npm install express --save
```

Install the Body-Parser Middleware

```
npm install --save body-parser
```

Install View Engine for Express

```
sudo npm install --save hbs
```

Install Serverless-HTTP to Connect API to AWS

```
npm install --save serverless-http
```

Once done, open up the handler.js file from your serverless app directory, clear it and paste in the following code:

Create a Website

Now we can create Html files, javascript files, css files, basically anything we want. *Let's do it.*

Create views

Create views folder in your serverless app directory and so that we add some files to it;

```
mkdir views
```

Your project should look similar to this;

Create index.hbs

Create a index.hbs file in views folder in your serverless-app directory;

```
vim index.hbs
```

Here's the code I'm adding to index.hbs file;

Create a Service

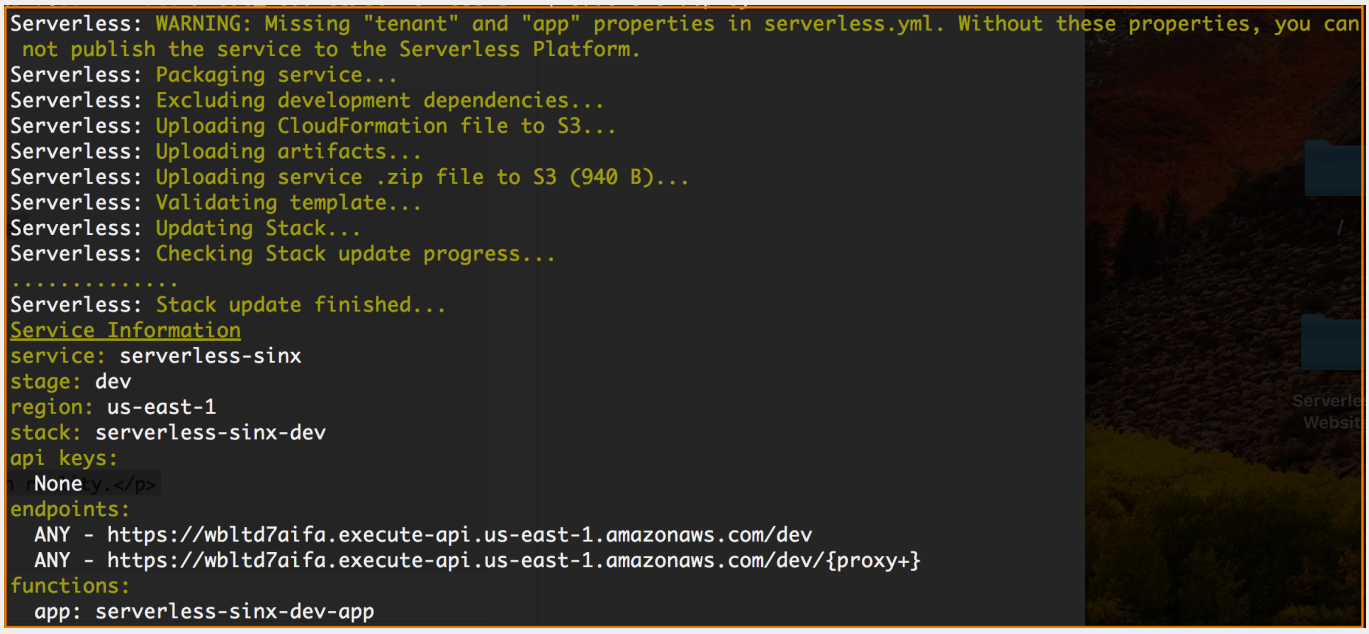
Let’s go ahead make some changes to serverless.yml file. We are writing it from scratch, so open it and add the following code;

Deployment

Now the final step is to deploy our code to AWS, so that we can have your own serverless website up and running. Now in your terminal run;

```
sls deploy
```

Once done, you’ll see something like this:



Deploy Serverless Website to AWS Lamda

In less than minute it will return an endpoint that we can open up in a browser.

> *You can now go ahead and boldly Say “Hello World”*

That’s it!

You’ve successfully deployed your first Serverless Website.

If you liked this article, I’ve got a few very practical reads for you. One about the AWS Serverless Courses and one about AWS Certified Solutions Architect exam.

> I’ve also got this data-centric newsletter that you might be into. I send a tiny email once every fortnight (if that) with some useful and cool stuff I’ve found/made.
> Don’t worry, I hate spam as much as you.