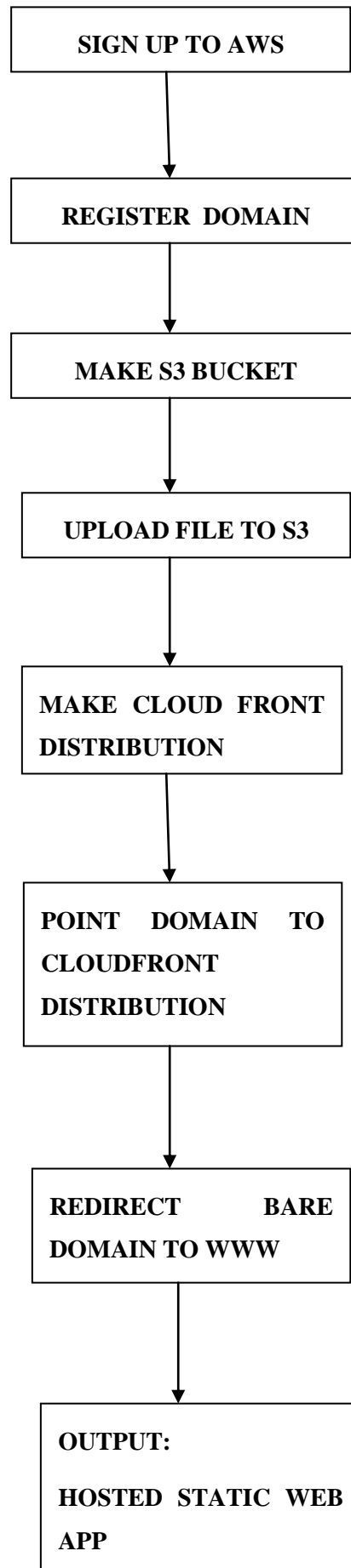


HOST A STATIC SITE ON AWS, USING S3, ROUTE 53 AND CLOUDFRONT

ABSTRACT

In our project we are going to create a static website and making it to be getting accessed on the internet by using the Amazon services. There are many different ways to create a static website, but we are using Lektor. It's a static content management system that makes creating a static website a breeze. Once it's set up, it's easy to create, edit, and delete content from your website. We are using the S3, Route53, cloudfront services of Amazon.

WORK FLOW DIAGRAM



STEPS TO BE FOLLOWED WITH SNAPSHOTS.

- **STEP 1: SIGN UP WITH AWS**

Amazon Web Services, or AWS, is the most popular cloud computing environment today. It has many different services that can be used together or separately, and because we only pay for what you use, it's very inexpensive to get started. The first step is to sign up for an AWS account, by going to aws.amazon.com and clicking on the button that says "Create a Free Account". If we already have an account with Amazon.com, we can put in the username and password that we use for Amazon.com, since Amazon.com and AWS are run by the same company. Once you've signed up with AWS, you can access the AWS Console at console.aws.amazon.com. In our we are using only three services S3, Route5, and cloudfront.

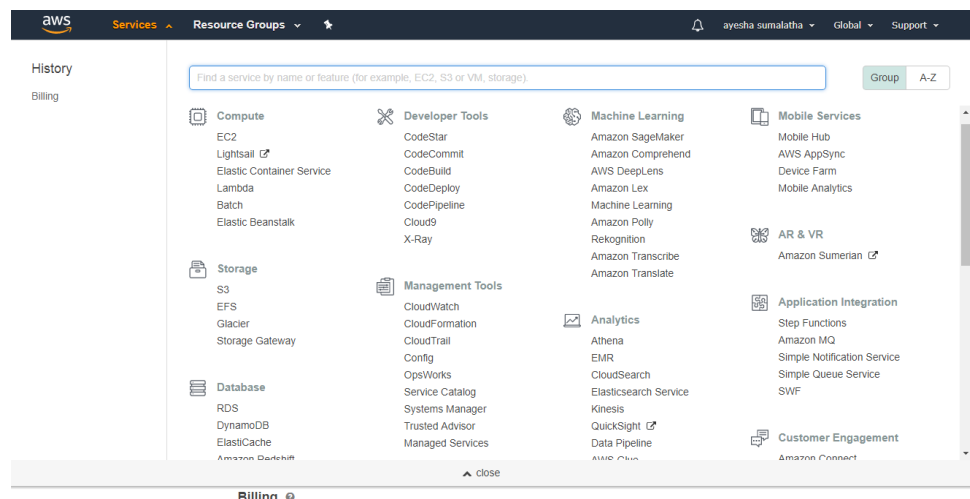


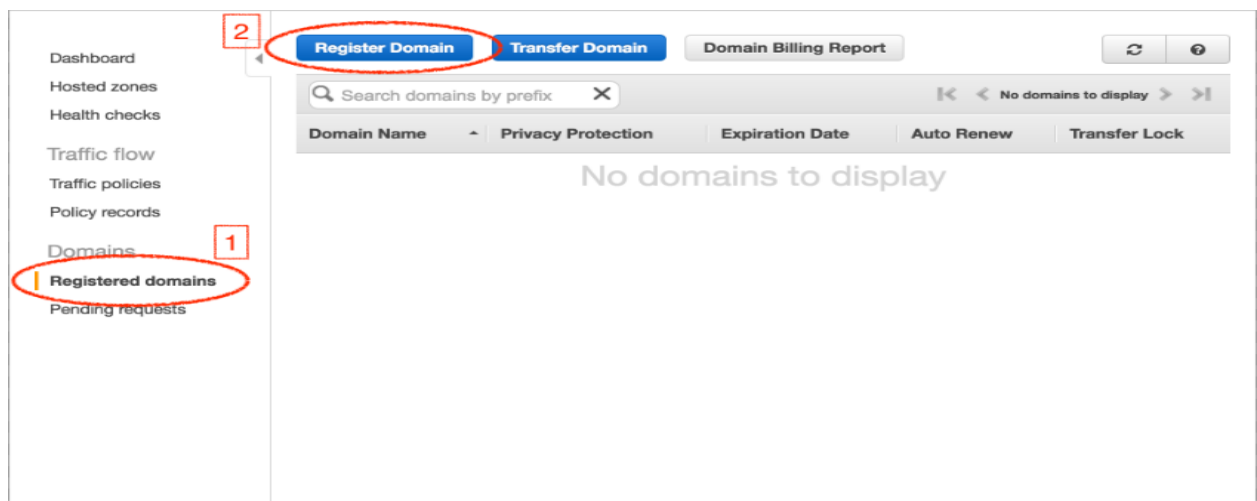
Fig1: AWS Services

STEP 2 : REGISTER A DOMAIN

To host our static site on AWS, the first thing we did is registering a domain for it. A domain is the first part of the URL that we type in to access your site, often ending in ".com". (That ".com" part is called a top-level domain, or TLD. There are many different TLDs, but ".com" is the most well-known.). "Registering" a domain basically means claiming ownership of it for a specified period of time, usually one to five years. There's a big, global network called the Domain Name System, or DNS, which determines who owns which domains. We cannot claim ownership of a domain forever, but we can claim ownership for several years at a time, and keep extending that claim over and over again as many times as

you want. That's how big companies keep their websites running for years at a time , but if we forgets to renew the domain before the ownership expires, it gets released and anyone else can register it.

In order to register a domain in the DNS,we go through a "regis, and you have to pay a fee to that registrar. Usually, this fee is around \$10 per year that we are claiming ownership of the domain, but different registrars charge different amounts. As we might expect, AWS is a domain registrar, and in order to register a domain with AWS,we are using the service called "Route 53".Go to the AWS Console and click on "Route 53", under the "Networking" section. Then go to the Domain Registration section, click on the "Register Domain" button, type in the domain we want to register, select the TLD that you want to use, and click "Check". The pages should look like this:



aws Services Resource Groups

ayesha Global Support

1: Domain Search

Choose a domain name

2: Contact Details

ayesumawebsite .com - \$12.00 **Check**

3: Verify & Purchase

To register a domain name, start by finding one that's available. Enter the first part of the name (such as example in example.com), choose an extension (such as .com or .org), and click Check. We'll tell you whether it's available and whether you can get it with other extensions. [Learn more](#).

Shopping cart

Cancel Continue

Feedback English (US) © 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

When we click on the "Check" button, Route 53 will check DNS to see if someone else has already claimed that domain. If so, we have to try a different name. There's a handy website called Domainr that help us to find variations on the name we were hoping for. Unfortunately, most short, simple domain names are already taken, so we probably need to get a bit creative.

Once we've found an available domain that we like, add it to our Route 53 shopping cart and click "Continue". On the next page, we can select how many years we want to register the domain for: the longer we claim it, the more it will cost. We are also required to put in contact information for whoever owns and controls this domain. If we are buying this domain to host a personal websit.

1: Domain Search

Contact Details for Your 1 Domain

2: Contact Details

3: Review & Purchase

Enter the details for your Registrant, Administrative and Technical contacts below. All fields are required unless specified otherwise. [Learn more](#).

My Registrant, Administrative and Technical Contacts are all the same: ☒ Yes ☐ No

Registrant Contact

Contact Type ⓘ Person

First Name

Last Name

Organization ⓘ Not applicable

Email

Phone + e.g. 1 e.g. 3115550188

Shopping cart

One-time fees

my-website.com

Register for 1 year \$12.00

SUBTOTAL \$12.00

Monthly Fees for DNS Management

[View pricing details](#) for Route 53 queries and for the hosted zone that we create for each new domain.

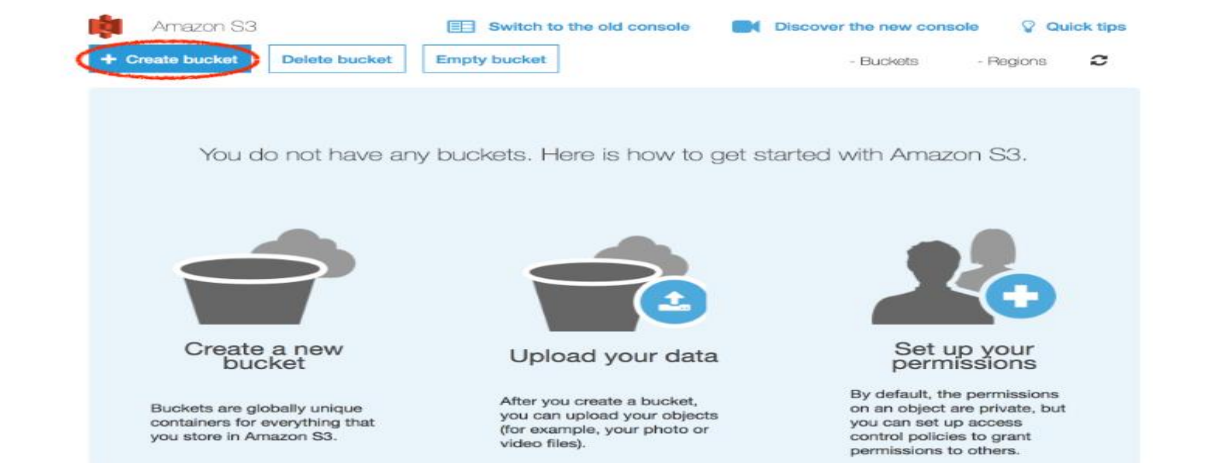
Note that **any contact information we put into this form will be publicly visible** unless we enable Privacy Protection to our domain, at the bottom of the contact form. We are putting our own contact information, be sure to enable Privacy Protection if you don't want everyone in the world knowing your home address.

When we've put in the contact information, click Continue. Review your information, check the "Terms and Conditions" check box, and click "Complete Purchase". Well done, we now own our first domain.

STEP3: MAKE AN S3 BUCKET

The domain is basically a pointer that tells computers how to get to ur website, but we don't have a place to point those computers to yet. That's where S3 comes in. S3 stands for "Simple Storage Service", and it's going to hold the HTML files that make up our static website. Conceptually, we're going to deploy our website in three parts. S3 will hold the files that make up your website, CloudFront will serve those files out to the internet, and our domain will point to CloudFront so that people can find our website on the internet. Let's take it one step at a time, and start with S3.

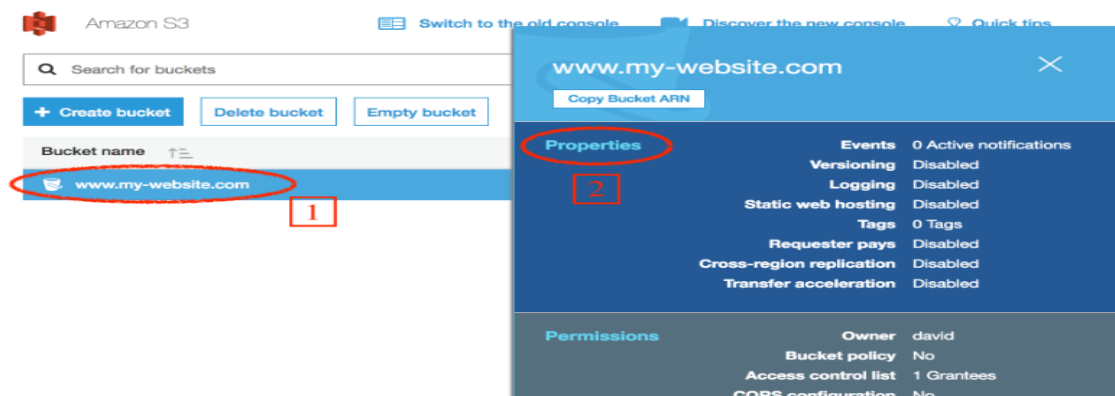
In order to use S3, we have created something called a "bucket". Like a literal bucket, it exists to hold things. S3 uses these buckets to organize different kinds of data and keep them separate from each other. we can use different S3 buckets to hold different kinds of data. We have lots and lots of buckets if we want, and the buckets themselves are totally free. Go to the AWS Console and click on "S3", under the "Storage & Content Delivery" section. Then click on the "Create Bucket" button. It should look like this:



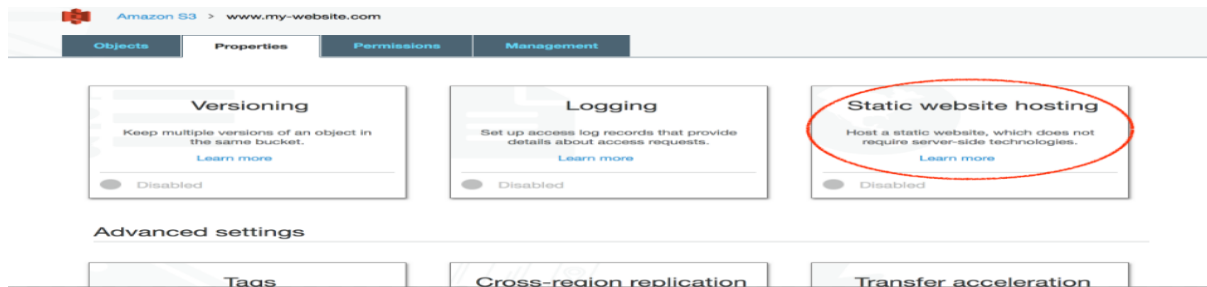
S3 will ask us to pick a name for our bucket. **This is important:** we *must* name our bucket with the exact domain name that we want our site to be available at. we want our website to be available at www.ayesumawebsite.co.uk, and we own the ayesumawebsite.com.uk domain, then the bucket must be named www.ayesuma.co.uk precisely.

Note that it is not strictly required to include the "www" at the front of your URL, but it's a good idea if you think there's any chance that our website will grow to a massive size in the future. There are arguments for the www and against the www, but here we will assume that you are using the "www".we also need to select a region for our bucket. This doesn't really matter, but I recommend that you select "US East (N. Virginia)" because it's the cheapest and most popular option.

Click the "Create" button, and S3 will create our bucket Next, we need to modify some properties on the bucket to make it hold our website correctly. Click on the row of the bucket we just created, and the details pane will open on the right side of the page. Then click on the "Properties" section of the page to modify the bucket properties. It should look like this:



On the properties page, click on the "Static Website Hosting" section to tell S3 that we want to use this bucket to host a website.



we need to set names for our index document and our error document; the default values of "index.html" and "error.html" should do fine, so type them in, and then click the "Save" button.

The image shows a modal window titled "Static website hosting" with a close button (X) in the top right corner. Inside the modal, the "Endpoint" is displayed as "http://www.my-website.com.s3-website-us-east-1.amazonaws.com". Below this is a radio button labeled "Use this bucket to host a website" with an information icon and a "Learn more" link. Underneath are two input fields: "Index document" with the value "index.html" and "Error document" with the value "error.html". There is also a section for "Redirection rules (optional)" with an information icon and a text area. At the bottom, there are two radio buttons: "Redirect requests" and "Disable website hosting", both with information icons and "Learn more" links. The "Save" button is highlighted in blue at the bottom right, next to a "Cancel" button.

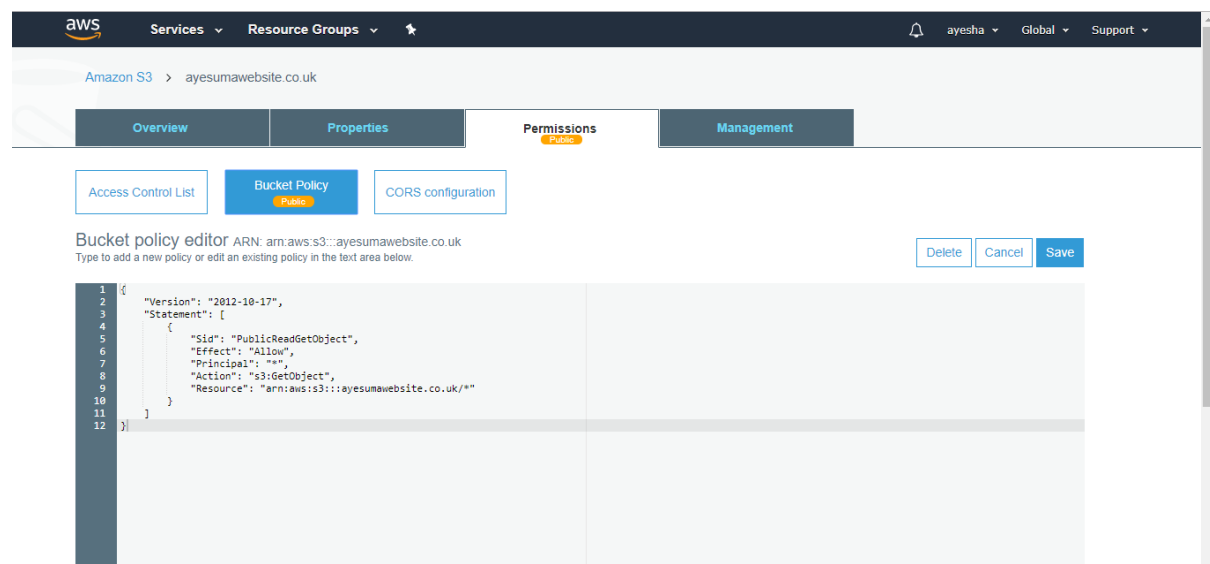
Do you see the "Endpoint" URL at the top of this form? It should look something like <http://ayesuma.co.s3-website-us-east-1.amazonaws.com>. This URL is very important, and we'll come back to it later.

Next, we need to change the permissions on this bucket so that all the files in this bucket can be viewed by anyone. To do this, we're going to use something called a "bucket policy", which tells S3 how you want it to treat the contents of the bucket.

After we save the static website hosting form, scroll back up to the top of the page and click on the "Permissions" tab, to the right of the "Properties" tab. Then, click on the big "Bucket Policy" button, right below the tabs. You'll see a big text box, marked "Bucket policy editor". Copy-paste this text into the bucket policy editor:

```
{
  "Version": "2008-10-17",
  "Statement": [{
    "Sid": "AllowPublicRead",
    "Effect": "Allow",
    "Principal": {"AWS": "*"},
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3:::www.ayesumawebsite.co.uk/*"]
  }]
}
```

In the line that starts with "Resource", we need to replace `www.my-website.com` with the name of our bucket. Then click the "Save" button. It should look like this:



Note that Amazon might reformat the bucket policy, so it has different spacing. Notice how the spacing in the screenshot is different from the spacing of the text above. That's totally fine the spacing of the bucket policy doesn't matter at all.our S3 bucket is all set to host a website, In fact, that "Endpoint" URL that you saw earlier is one way that people will be able to

access your site. However, there are two problems. One, this URL is very long and ugly: `http://www.my-website.com.s3-website-us-east-1.amazonaws.com` is not exactly a memorable name. And two, our bucket still doesn't have any HTML files in it. Fortunately, both problems can be fixed.

STEP4: UPLOAD TO S3

The next step is to upload our HTML files to S3. In our project, we'll assume we're using Lektor to generate your HTML files. If we're using Lektor, we need to configure Lektor to upload our HTML files to S3 every time we re-deploy our website. Fortunately, the `lektor-s3` plugin makes this a lot easier, but we still have to set it up properly, so that it works with the bucket we just created.

First, add the plugin to our project by running this command while sitting inside your Lektor project directory:

```
$ lektor plugins add lektor-s3
```

Next, open up our `.lektorproject` file in a text editor and add a new section to hold information about the S3 bucket we created. It should look like this:

```
[servers.s3]
name = S3
enabled = yes
default = yes
target = s3://www.my-website.com
```

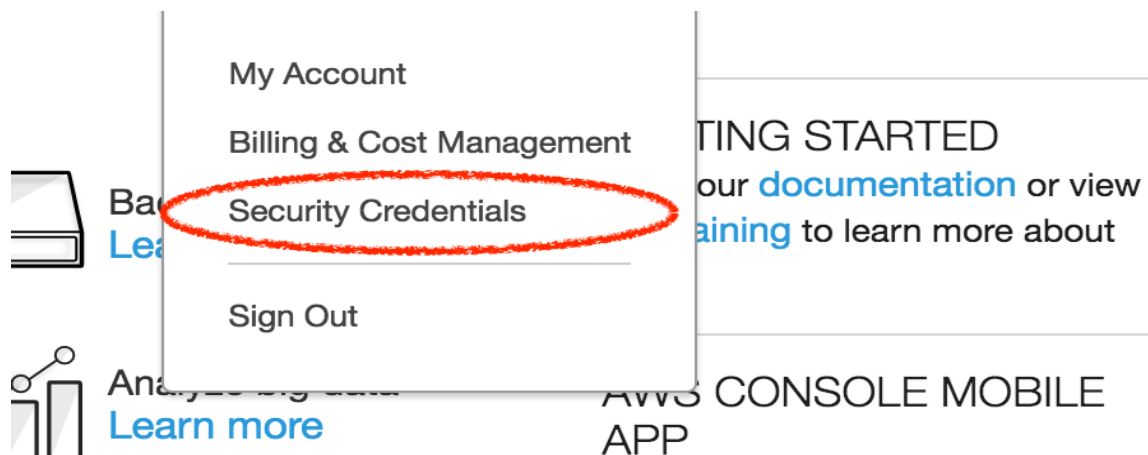
we need to change the target to refer to our bucket name, rather than leaving it as www.ayesumawebsite.co.uk. Don't forget to keep the `s3://` in front of the bucket name.

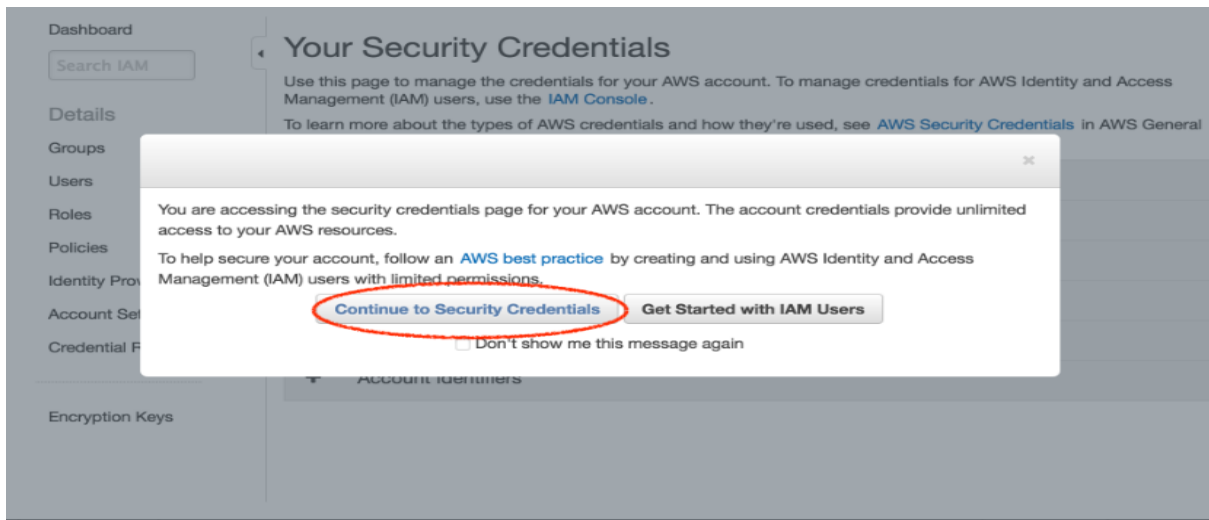
Next, we need to find your AWS credentials. "Credentials" just means "proof that we are who we say we are", and for many websites, we can use our username and password as credentials. However, AWS does things differently, and requires that us use an "Access Key

ID" and a "Secret Access Key", instead. we can generate multiple sets of credentials, and each set can have different permission levels, which can make our AWS account more secure. However, for the sake of simplicity, this project won't go into generating new sets of credentials, and instead will stick with our default, full-permissions credentials.

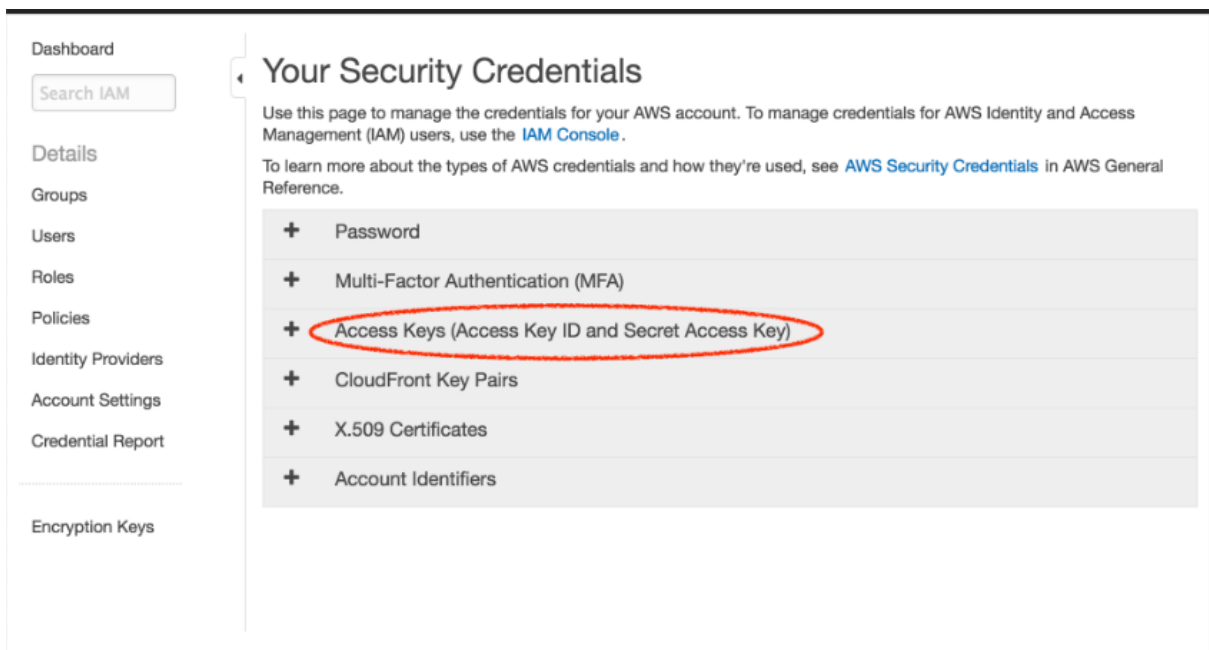
At the top right corner of every page of the <https://console.aws.amazon.com> AWS Console, our name will appear with a little arrow next to it. Clicking on our name will reveal a dropdown menu with only a few items. One of those items is "security credentials", which is what you want to click on.

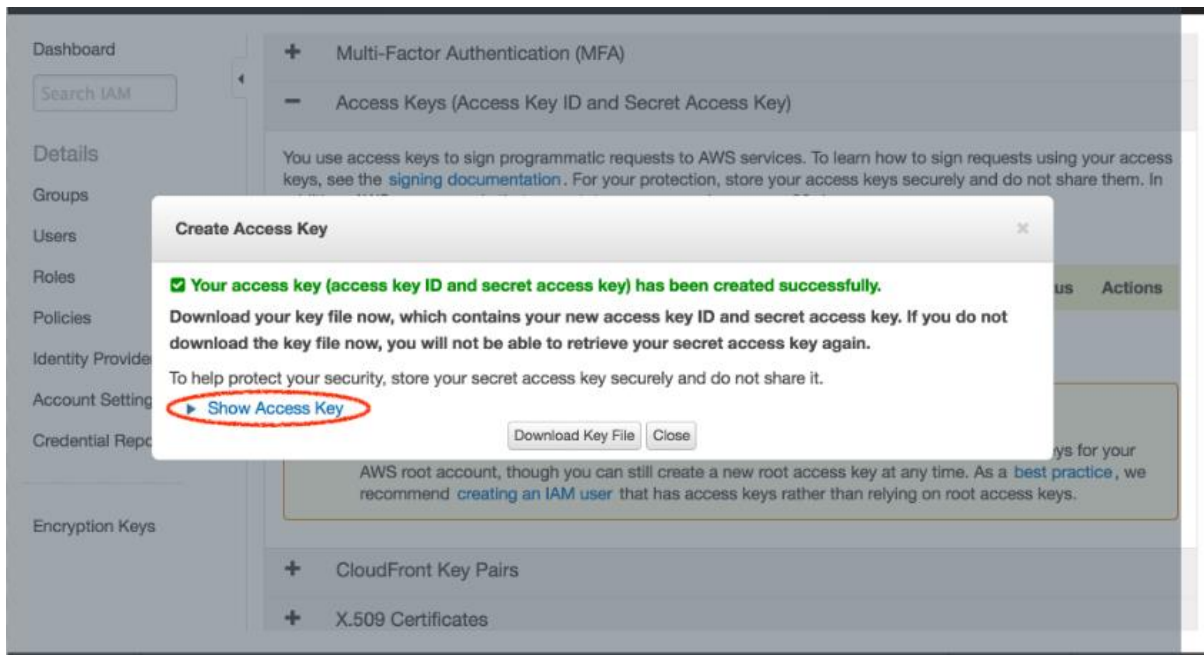
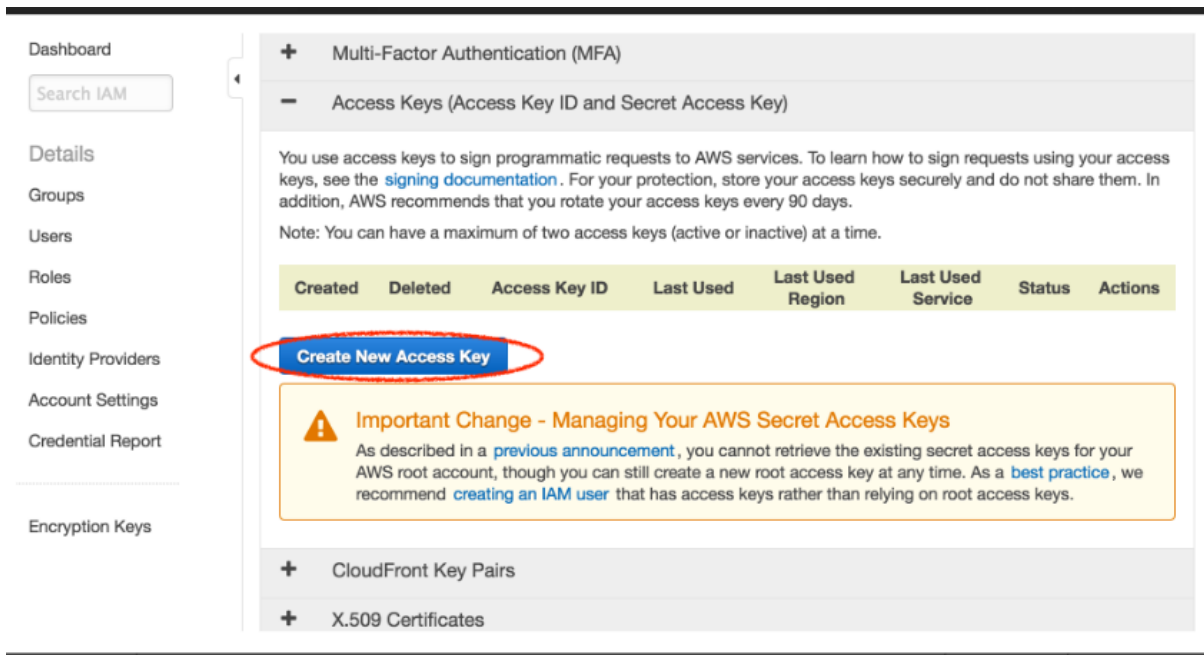
We will end up on a page that lists several options for security credentials, and a warning will immediately pop up, informing us that we should create users with limited permissions. That is beyond the scope of this project, so dismiss the warning by clicking on "Continue to Security Credentials".





Click on the "Access Keys" item in the list, and we can see a list of your existing access keys. If we have never used AWS access keys before, this list will be empty. Click on the "Create New Access Key" button, then click on "Show Access Key" once the access key has been created. The "Access Key ID" will be a long jumble of letters and numbers, and the "Secret Access Key" will be an even longer jumble of letters and numbers.





This is important: once we leave this page, we will never be able to download or view the secret access key again. Make sure that we save it to your computer. If we don't, and we need it later on, we have to delete it and create a new one instead. Now we need to save this information to a place on our computer that `lektor-s3` can use it. In our home directory, create a directory called `.aws`. Inside of that `~/.aws` directory, create a file named `credentials`, with the following contents:

```
[default]
aws_access_key_id=MYACCESSKEYID
```

```
aws_secret_access_key=MySecretAccessKey12345
```

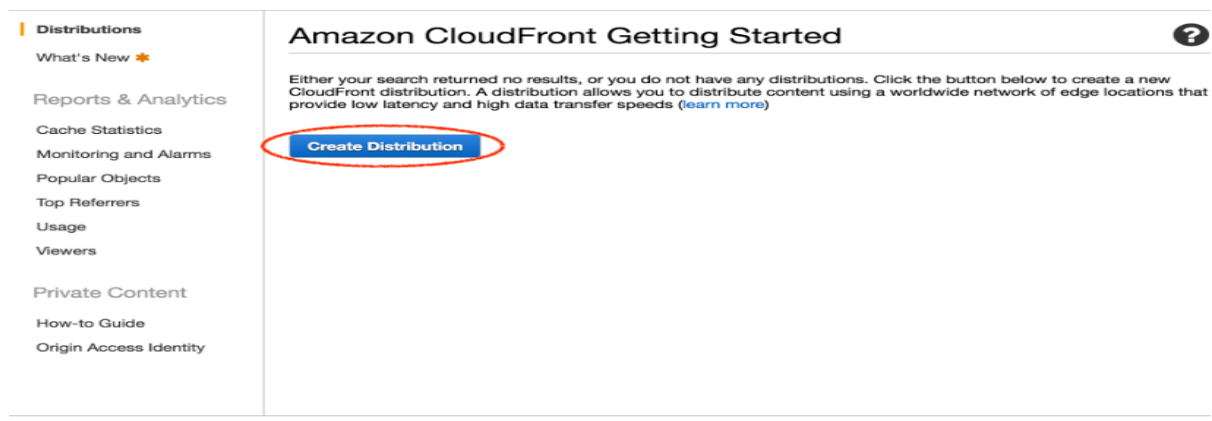
we need to change these values to refer to your Access Key ID and Secret Access Key, instead of using the example values here. Once we've created the `~/.aws/credentials` file with the correct contents, Lektor should be able to deploy our website's HTML files to the S3 bucket we created. Give it a try, by cd-ing into your Lektor project, and running:

Once it's finished, go back to the AWS Console, click on "S3", and click on our bucket. we should see several files and folders there, If we click on the "Properties" button in the toolbar at the top, click on the "Static Website Hosting" section, and click on the "Endpoint" URL, we will see our website.

STEP5: MAKE A CLOUDFRONT DISTRIBUTION

We've made great progress, but we're not done yet. The next step is to use CloudFront, Amazon's Content Delivery Network (CDN) service. This will do two things: make your website load *fast* anywhere in the world, and allow you to use HTTPS, the secure version of HTTP. Just like S3 is organized into "buckets", CloudFront is organized into "distributions". We need to create a distribution for our website and point it at your S3 bucket, so that it gets the same contents as our S3 bucket. Once that's up and running, we'll go back to Route 53 and make the domain you bought point to the CloudFront distribution, so that people who type in our URL can get to our website.

Go back to the AWS Console and click on "CloudFront", under the "Storage & Content Delivery" section. Click on the "Create Distribution" button to get started, and select that we want a Web distribution, instead of RTMP.



Step 1: Select delivery method
Step 2: Create distribution

Select a delivery method for your content.

Web

Create a web distribution if you want to:

- Speed up distribution of static and dynamic content, for example, .html, .css, .php, and graphics files.
- Distribute media files using HTTP or HTTPS.
- Add, update, or delete objects, and submit data from web forms.
- Use live streaming to stream an event in real time.

You store your files in an origin — either an Amazon S3 bucket or a web server. After you create the distribution, you can add more origins to the distribution.

Get Started

RTMP

Create an RTMP distribution to speed up distribution of your streaming media files using Adobe Flash Media Server's RTMP protocol. An RTMP distribution allows an end user to begin playing a media file before the file has finished downloading from a CloudFront edge location. Note the following:

- To create an RTMP distribution, you must store the media files in an Amazon S3 bucket.
- To use CloudFront live streaming, create a web distribution.

Get Started

The next page will be a *long* form where we can specify many different options for your distribution. Don't worry, we can leave most of these options at their defaults, but there's a few we need to change. The first one is the "Origin Domain Name" it's the very first field in the form. Click on it, and Amazon will show a dropdown menu with our S3 buckets listed. Unfortunately, this dropdown menu is misleading: the URL in the dropdown menu is *not* the URL that we want to use.¹ Instead, go back to our S3 bucket, find the website endpoint URL in the "Static Website Hosting" section of the properties, and paste that into the "Origin Default Name" field. we need to remove the `http://` from the front. It should look something like www.ayesiuma.com.s3-website-us-east-1.amazonaws.com.

Step 1: Select delivery method
Step 2: Create distribution

Create Distribution

Origin Settings

Origin Domain Name

Origin Path

Origin ID

Origin Custom Headers	Header Name	Value
	<input type="text"/>	<input type="text"/>

Default Cache Behavior Settings

Path Pattern	Default (*)
Viewer Protocol Policy	<input checked="" type="radio"/> HTTP and HTTPS <input type="radio"/> Redirect HTTP to HTTPS <input type="radio"/> HTTPS Only
Allowed HTTP Methods	<input checked="" type="radio"/> GET, HEAD <input type="radio"/> GET, HEAD, OPTIONS

Step 1: Select delivery method
Step 2: Create distribution

Create Distribution

Origin Settings

Origin Domain Name	<input type="text"/>	?
Origin Path	<div> -- Amazon S3 Buckets -- www.my-website-us-east-2.s3.amazonaws.com -- Elastic Load Balancers -- No Origins Available </div>	?
Origin ID	No Origins Available	?

Origin Custom Headers	Header Name	Value
	<input type="text"/>	<input type="text"/>

Default Cache Behavior Settings

Path Pattern	Default (*)	?
Viewer Protocol Policy	<input checked="" type="radio"/> HTTP and HTTPS <input type="radio"/> Redirect HTTP to HTTPS <input type="radio"/> HTTPS Only	?
Allowed HTTP Methods	<input checked="" type="radio"/> GET, HEAD <input type="radio"/> GET, HEAD, OPTIONS <input type="radio"/> GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE	?



Endpoint : <http://ayesumawebsite.co.uk.s3-website-us-east-2.amazonaws.com>

☒ Use this bucket to host a website [?](#) [Learn more](#)

Index document [?](#)

Error document [?](#)

Redirection rules (optional) [?](#)

☐ Redirect requests [?](#) [Learn more](#)

☐ Disable website hosting

Cancel

Save

aws Services Resource Groups

Step 1: Select delivery method
Step 2: Create distribution

Create Distribution

Origin Settings

Origin Domain Name

Origin Path

Origin ID

Restrict Bucket Access ☐ Yes ☒ No

Origin Custom Headers

Header Name	Value
<input type="text"/>	<input type="text"/>

Default Cache Behavior Settings

Path Pattern

Viewer Protocol Policy ☒ HTTP and HTTPS ☐ Redirect HTTP to HTTPS ☐ HTTPS Only

Allowed HTTP Methods ☒ GET, HEAD

Feedback English (US) © 2008 - 2018, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Next, scroll down to the "Distribution Settings" section. Just above it, there's a field called "Compress Objects Automatically": set that to "Yes". Slightly below that, there's a field called "Alternate Domain Names (CNAMEs)". Put the domain of your website into the CNAMEs field: for example, "www.ayesuma.com". This should match the name of the S3 bucket we're using. Finally, scroll down to the bottom of the form, find the "Default Root Object" field, and type in "index.html". Then click "Create Distribution".

Step 1: Select delivery method
Step 2: Create distribution

Restrict Viewer Access ☐ Yes ☒ No (Use Signed URLs or Signed Cookies)

Compress Objects Automatically ☒ Yes ☐ No [Learn More](#)

Distribution Settings

Price Class

AWS WAF Web ACL

Alternate Domain Names (CNAMEs)

SSL Certificate ☒ Default CloudFront Certificate (*.cloudfront.net)

Choose this option if you want your users to use HTTPS or HTTP to access you with the CloudFront domain name (such as <https://d1111111abcdef8.cloudfront.net/logo.jpg>). Important: If you choose this option, CloudFront requires that browsers or device support TLSv1 or later to access your content.

☐ Custom SSL Certificate (example.com):

Step 1: Select delivery method
Step 2: Create distribution

Request an ACM certificate

[Learn more](#) about using custom SSL/TLS certificates with CloudFront.
[Learn more](#) about using ACM.

Default Root Object

index.html

Logging

☐ On
☒ Off

Bucket for Logs

Log Prefix

Cookie Logging

☐ On
☒ Off

Comment

Distribution State

☒ Enabled
☐ Disabled

Cancel

Back

Create Distribution

AWS will start creating our CloudFront distribution, which takes a few minutes to complete. we can see that this screenshot shows that it's still in progress.

Distributions
What's New ✨
Reports & Analytics
Cache Statistics
Monitoring and Alarms
Popular Objects
Top Referrers
Usage
Viewers
Private Content
How-to Guide
Origin Access Identity

CloudFront Distributions

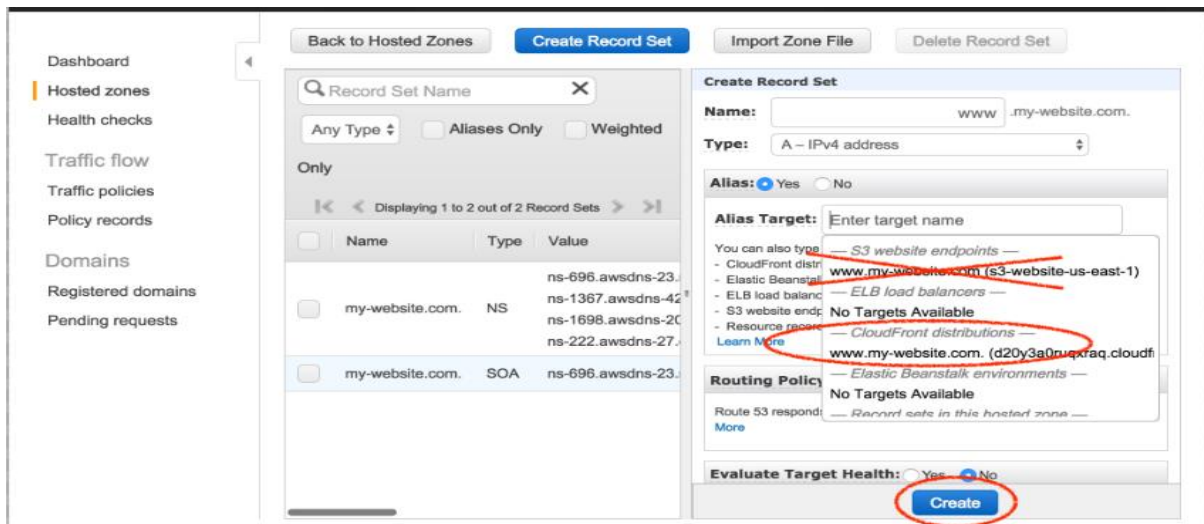
Create Distribution
Distribution Settings
Delete
Enable
Disable

Viewing : Any Delivery Method Any State

	Delivery	ID	Domain Name	Comr	Origin	CNAI	Status	State	Last
<input type="checkbox"/>	Web	E19C	d3lg2kbp01o9f.cloudfront.net	-	www.	www.	In Progress	Enabled	2016

STEP6: POINT DOMAIN TO CLOUDFRONT

Now that we have our website up and running on both S3 and CloudFront, the next step is configuring that domain you bought to point at it. To do this, we have to go back into Route 53, so go back to the AWS Console and click on Route 53. Then click on your domain, and click "Manage DNS".

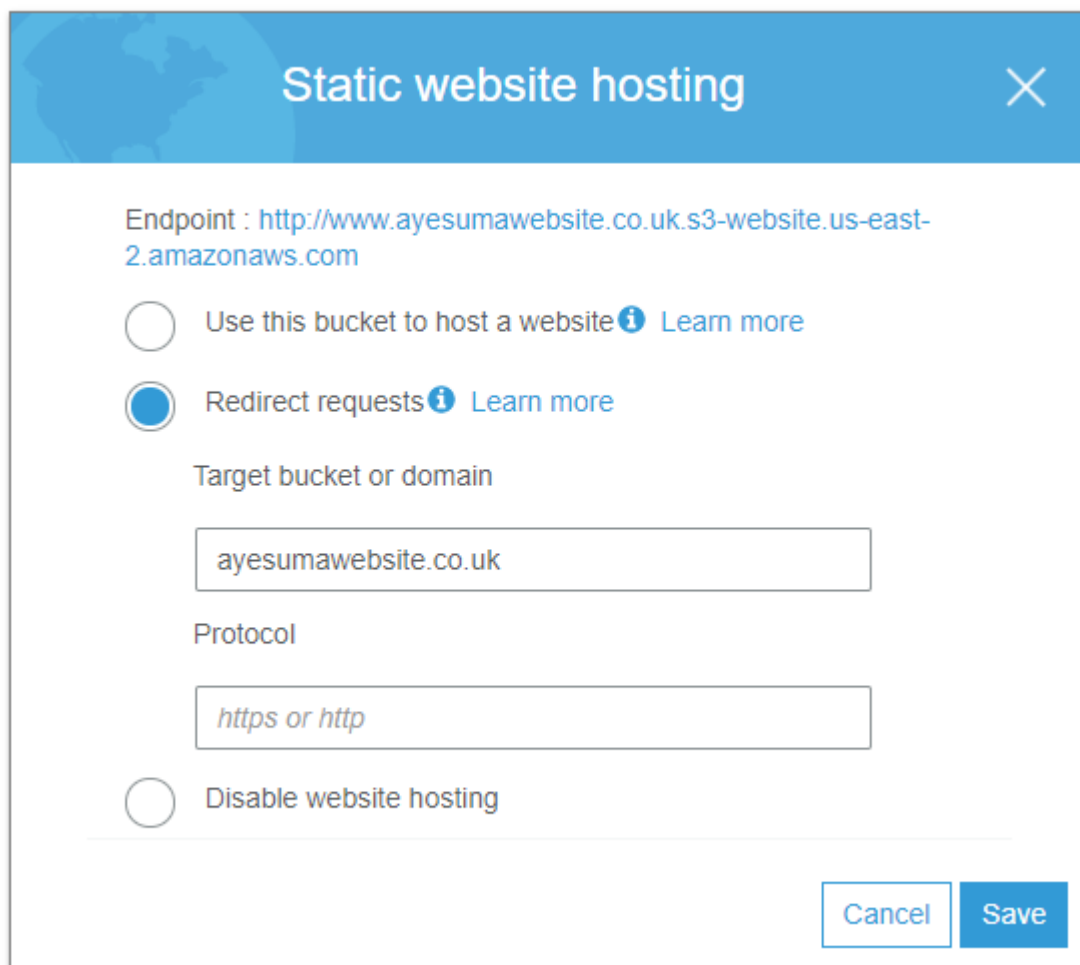


When we click on the "Alias Target" field, Amazon will show another helpful dropdown menu, with entries for both S3 and CloudFront. Select the CloudFront distribution we just created, *not* the S3 bucket. Then click the "Create" button. Whenever we make changes to DNS, those changes don't apply immediately. DNS uses a technique called *cacheing* to avoid overloading the system, and one of the downsides of caching is that it can take a little while for the system to notice changes. When we make a DNS change, it can take anywhere from 5 minutes to 1 hour for it to apply.

Type our domain into our web browser, with the "www" in front, and see if our website shows up. If not, wait a little while and try again it shouldn't take longer than an hour. Once it works, celebrate. You have a website.

STEP 7: REDIRECT BARE DOMAIN TO WWW

Our website works, but only if we include the "www" in front if we try to visit our domain without the "www", it will fail, which is very confusing for we are trying to visit our site. A domain without a "www" is called a "bare domain", and the correct way to fix this is with something called a "bare domain redirect". With a bare domain redirect in place, if someone tries to visit our site without the "www", they will be automatically rerouted to the "www" domain, so that the website works without the user having to do anything. Setting this up is pretty straightforward, and it uses the same three services you've already used: S3, CloudFront, and Route 53.



The screenshot shows the 'Static website hosting' configuration window in the AWS console. The window has a blue header with the title 'Static website hosting' and a close button (X). Below the header, the 'Endpoint' is displayed as 'http://www.ayesumawebsite.co.uk.s3-website.us-east-2.amazonaws.com'. There are three radio button options: 'Use this bucket to host a website' (unselected), 'Redirect requests' (selected), and 'Disable website hosting' (unselected). Each option has an information icon (i) and a 'Learn more' link. Below the 'Redirect requests' option, there are two text input fields: 'Target bucket or domain' containing 'ayesumawebsite.co.uk' and 'Protocol' containing 'https or http'. At the bottom right, there are 'Cancel' and 'Save' buttons.

Static website hosting

Endpoint : <http://www.ayesumawebsite.co.uk.s3-website.us-east-2.amazonaws.com>

☐ Use this bucket to host a website [Learn more](#)

☒ Redirect requests [Learn more](#)

Target bucket or domain

Protocol

☐ Disable website hosting

[Cancel](#) [Save](#)

go back to the <https://console.aws.amazon.com>>AWSConsole and click on S3. Make a new S3 bucket named after our domain without the `www`. If our domain is `ayesuma.com`, then the bucket should be named `ayesuma.com`. Click on this newly-created bucket to open the details pane, click on the "Properties" section, and then click on the "Static Website Hosting" section on the bucket properties page. In the static website hosting form, select the "Redirect requests" option. For the "Target bucket or domain" field, put in the name of our other S3 bucket that does have the `www` in there. Then click the Save button to save your changes.

Next, we need to make a CloudFront distribution for this redirect bucket. Go back to the AWS Console and click on CloudFront. Create a new CloudFront distribution just like before, but this time, put the URL for our bare domain S3 bucket in the "Origin Domain Name" field. Remember, the dropdown menu for this field is misleading: copy-paste the website endpoint URL from the static website hosting form, instead of selecting the S3 bucket from the dropdown menu. We should also put the bare domain in the "Alternate

Domain Names (CNAMEs)" field. Everything else is the same: still a web distribution rather than RTMP, and we can still leave most of the fields at their default values.

Finally, we need to set up this CloudFront distribution in DNS. Go back to the AWS Console and click on Route 53. Add a new record set for our domain, but this time, leave the name blank. This is how Route 53 knows the record should apply to the bare domain, instead of the "www". Just as before, the record should be of type "A", with "Alias" set to "Yes". Select the CloudFront distribution that we just made for the bare domain, and then click "Create". Once again, this change will take anywhere from 5 minutes to 1 hour to take effect. Once it does, visiting the bare domain in our web browser will automatically redirect us to the "www" subdomain.