Exercise 1.2: Creating a S3 Bucket and Configuring as a Static Website

Exercise 1.2: Creating a S3 bucket and configuring as a static website

Now we have our chatbot fundamentals out of the way. I would like to put that to one side and revisit it in week 4.

What I'd like us to focus on next is the building of a database driven web application that figures out the temperature when we tell it a particular city.

As you would imagine, creating an application like this involves many moving parts.

We will need:

- a front end web site
- a backend API
- some sort of logic (functional intelligence)
- a data source.

If we take it stage by stage and first focus on just the web front end, and not concern ourselves yet with wiring it up to a backend. We only need to worry about hosting some HTML, CSS and little bit of JavaScript.

We already have a website front end all prepared for you in a zip file. I did promise you no coding remember :)

All you need to do is download this zip file, unzip it and push it into the cloud. We will use S3 (Simple Storage Service) which can happily "host" these kinds of static websites.

Later on, after we have this website uploaded and everything is configured for website hosting. We can get a little fancy and add a content delivery network in front of it. Ready for global domination :). Adam will talk to you about that in one of the upcoming videos.

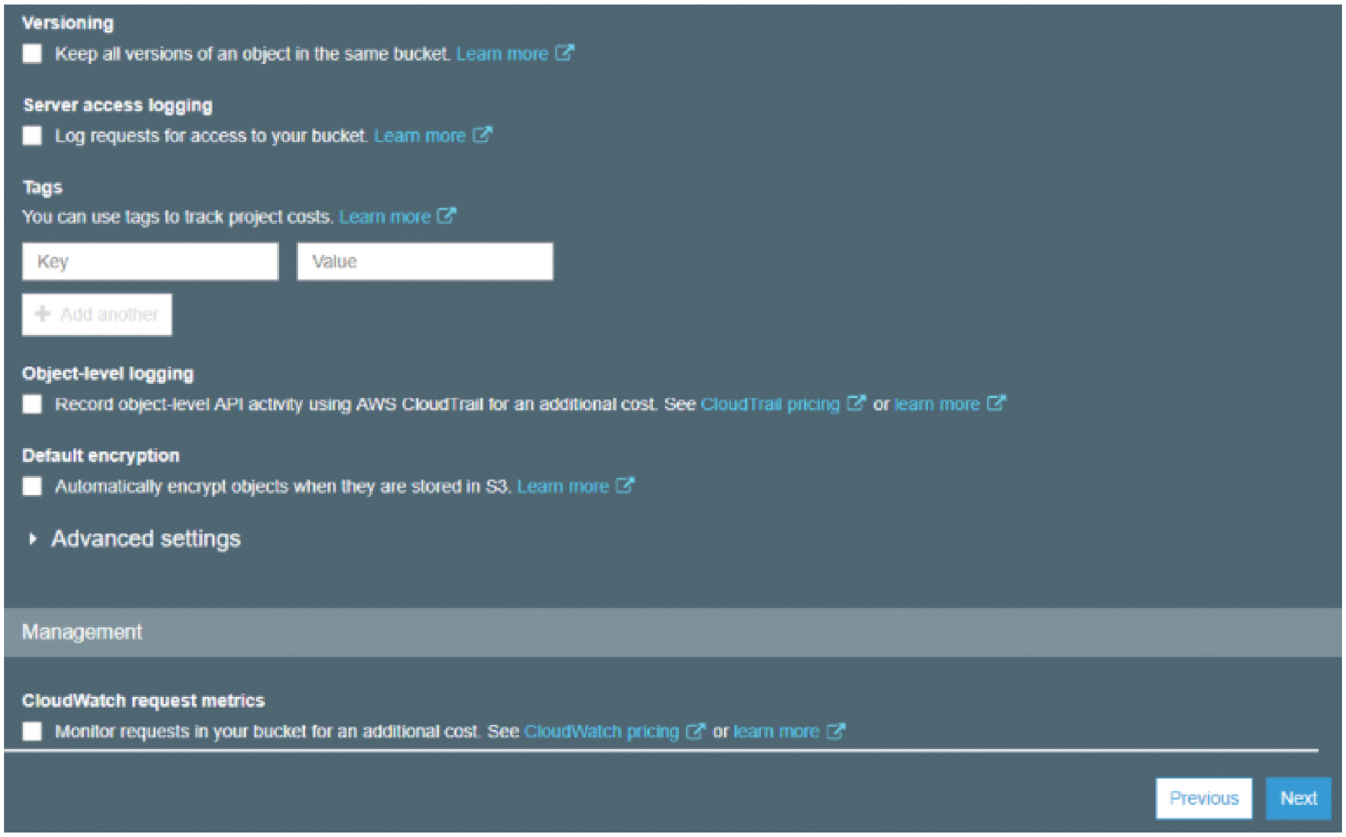
Al things in good time though, let's get this static website up and running on S3 first.

You'll need to create what we call a "bucket" to store your web objects (HTML, CSS, and JS files), and set it up correctly using a bucket policy so it can act as a host for out static website.

It's not difficult to do this, just follow these steps carefully, and your site will be up in no time.

1. Steps to create a S3 bucket and configure it as an static website.

- Sign in to the AWS Management Console and in the **Find Services** search box type s3 and choose **S3**.
- Click **Create bucket**.
    - In the **Bucket name** field, type a unique DNS-compliant name for your new bucket.
    - Example: `2019-03-01-er-website` *IMPORTANT: Use your own initials and the current date.*
    - For **Region** choose US East (N. Virginia).
- Click **Next**. Leave the current settings alone:



Versioning
☐ Keep all versions of an object in the same bucket. Learn more

Server access logging
☐ Log requests for access to your bucket. Learn more

Tags
You can use tags to track project costs. Learn more

Key | Value

Object-level logging
☐ Record object-level API activity using AWS CloudTrail for an additional cost. See CloudTrail pricing or learn more

Default encryption
☐ Automatically encrypt objects when they are stored in S3. Learn more

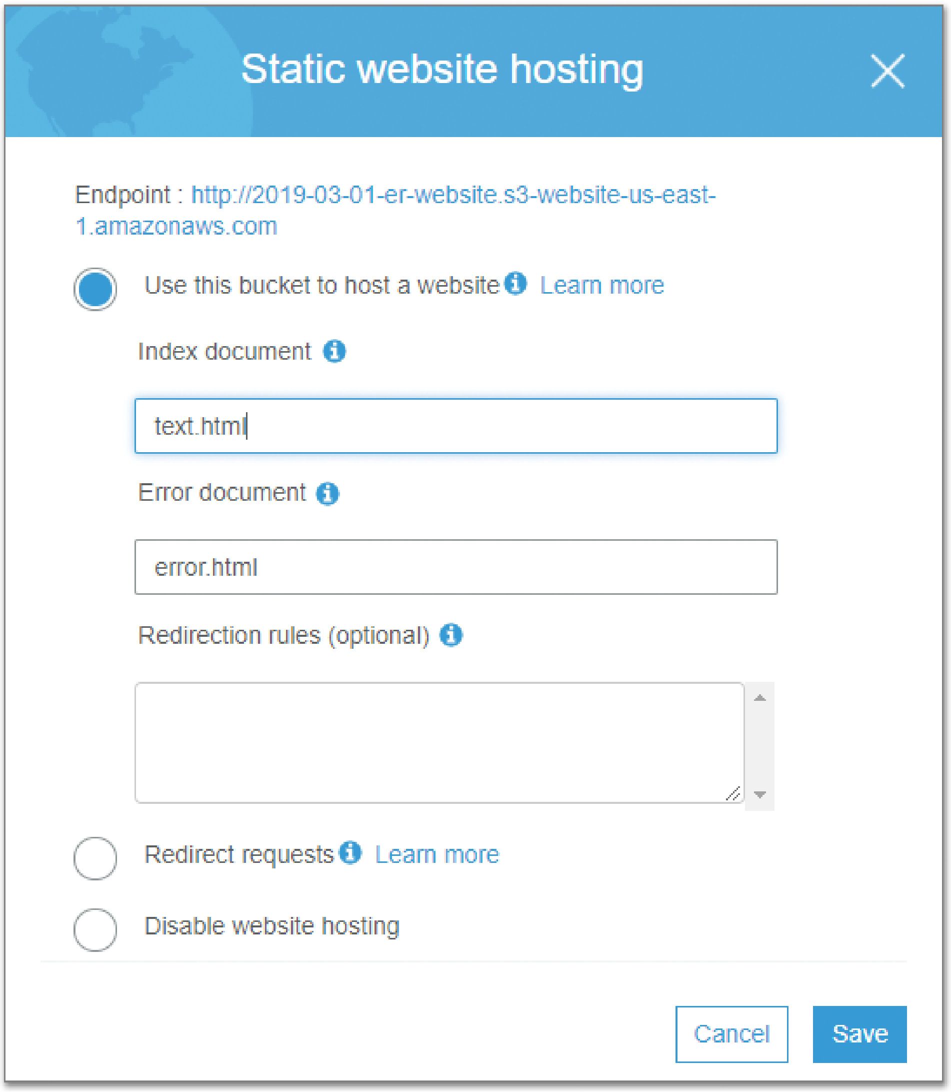
Advanced settings

Management

CloudWatch request metrics
☐ Monitor requests in your bucket for an additional cost. See CloudWatch pricing or learn more

- Click **Next** again.
- Under **Public access settings for this bucket** de-select the following:
    - **Block new public ACLs and uploading public objects**.
    - **Remove public access granted through public ACLs**.
    - **Block new public bucket policies**.
    - **Block public and cross-account access if bucket has public policies**.

- Click **Next**.
- Click **Create bucket**.
- Click on the bucket and select the **Properties** pane, choose **Static Website Hosting**.
  - Select **Use this bucket to host a website**.
  - Under **Index document** type in `text.html`. *NOT index.html*
  - Under **Error document** type in `error.html`.



Endpoint : http://2019-03-01-er-website.s3-website-us-east-1.amazonaws.com

[✓ Use this bucket to host a website] Learn more

Index document: text.html

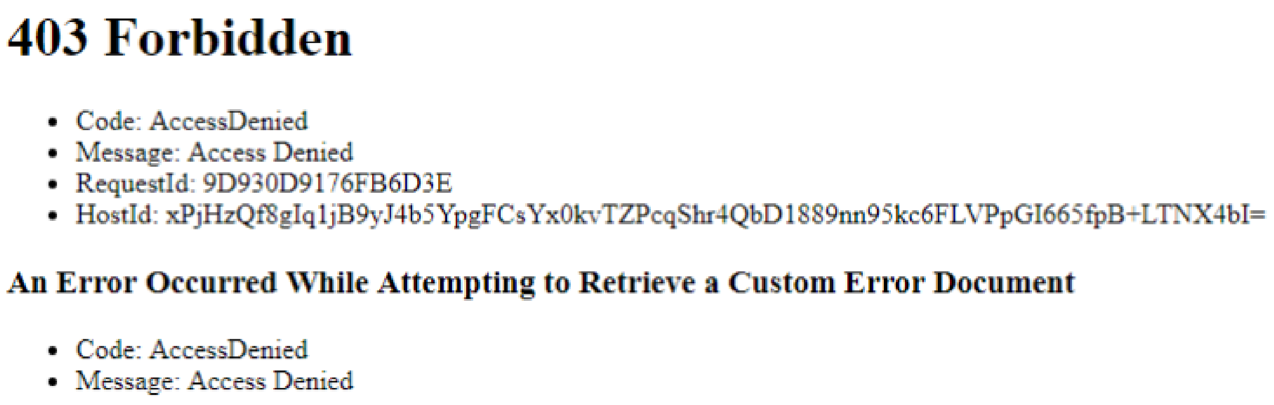
Error document: error.html

Redirection rules (optional):

Redirect requests Learn more

Disable website hosting

- Click the endpoint. It should open in a new tab and show an error 403 forbidden. Which is exactly what we would expect right now, as no permissions have been added yet.



403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: 9D930D9176FB6D3E
- HostId: xPjHzQf8gIq1jB9yJ4b5YpgFCsYx0kvTZPcqShr4QbD1889nn95kc6FLVPpGI665fpB+LTNX4bI=

An Error Occurred While Attempting to Retrieve a Custom Error Document

- Code: AccessDenied
- Message: Access Denied

- Click **Save**. So we can move on to adding public permissions to this bucket.
    - Select the **Permissions** tab and choose **Bucket Policy**.
    - Paste the following code below in the Bucket policy editor. Make sure to change the Resource arn with your bucket name.
    - You must change `arn:aws:s3:::2019-03-01-er-website/*` to your bucket. e.g `arn:aws:s3:::2019-mm-dd-xx-website/*`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AddPerm",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::2019-03-01-er-website/*"
            ]
        }
    ]
}
```

- Click **Save**.
- This bucket will now have public access as it will host our web objects.

Try it now. Go to that tab where your website is hosted (where we had the 403 forbidden message), and press refresh.

You will notice you can now see absolutely nothing. i.e you will get a 404 error.

This is because we have nothing in out bucket to show, so the browser is telling us, "nope can't find anything here". At least this proves that the bucket is now public and we did everything right so far #winning.

Now let's get the website up there.

2. Steps to upload objects to an existing S3 bucket.

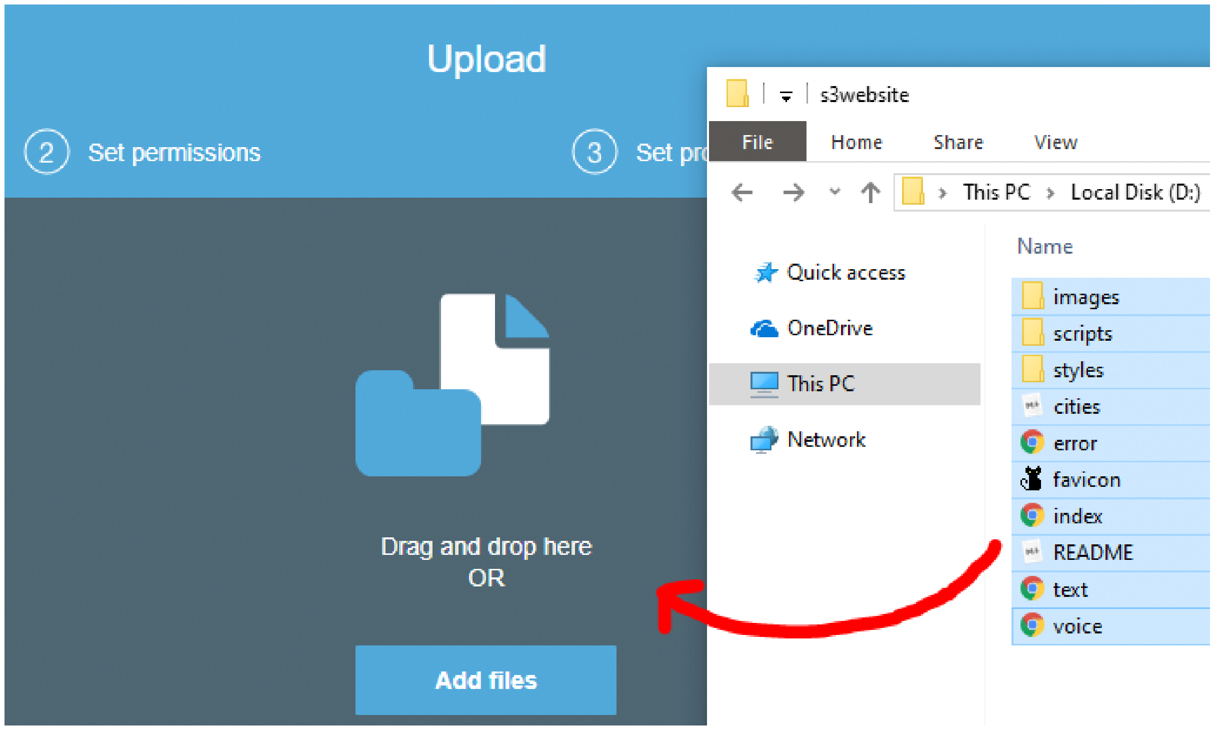
- Download the ZIP file at:

```
https://s3.amazonaws.com/awsu-hosting/CSA-TF-100-SCSRVL-10-EN/s3website.zip
```

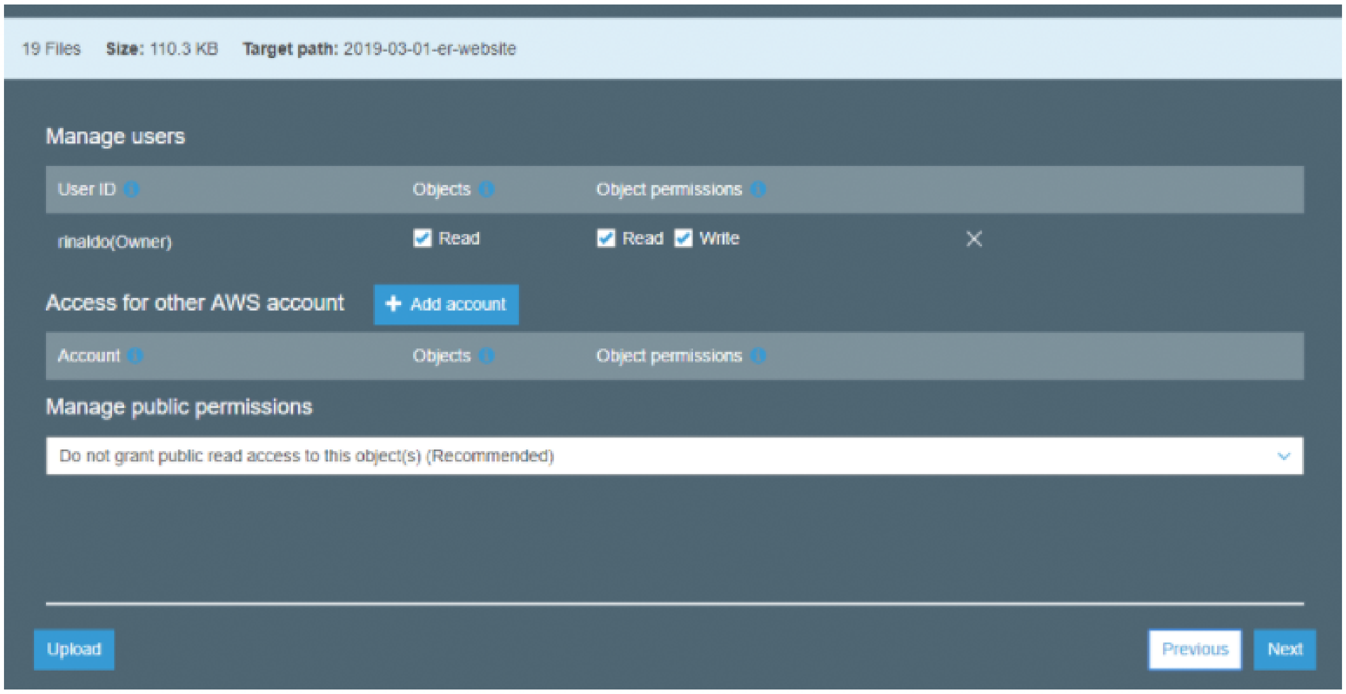
- Unzip it on your local machine, and you will see the following folder structure:

```
+ images
    - mobile.jpg
+ scripts
    - config.js
    - helper.js
    - jquery.js
    - text.js
    - voice.js
+ styles
    - reset.css
    - text.css
    - voice.css
+ cities.md
+ error.html
+ favicon.ico
+ index.html
+ README.md
+ text.html
+ voice.html
```

- Click the **Overview** tab and click on your bucket **2019-mm-dd-xx-website**. (Keep in mind your bucket will have its own unique name that you gave it).
- Click **Upload** and drag the files needed for the site from your local file explorer.
- TIP: You can press "add files" button, but we recommend dragging (like in this image below)



- Click **Next**.
- Leave the permissions settings alone as the defaults.


19 Files **Size:** 110.3 KB **Target path:** 2019-03-01-er-website

Manage users

User ID	Objects	Object permissions
rinaldo(Owner)	✓ Read	✓ Read ✓ Write

Access for other AWS account

Account	Objects	Object permissions

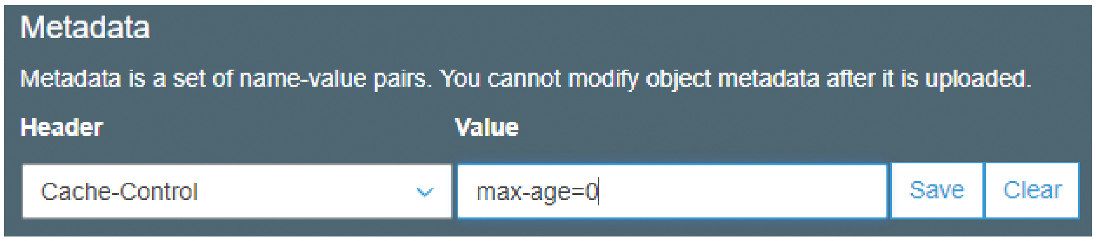
Manage public permissions

Do not grant public read access to this object(s) (Recommended)

- Click **Next**.
- Scroll down to **Metadata**.

⚠️This next step about max age is **critical!** If you miss this, you will have problems later on in the future exercises. So type carefully and do not miss this step.

- For **Header** choose `Cache-Control` and for the **Value** enter **max-age=0** and click **Save**.


Metadata

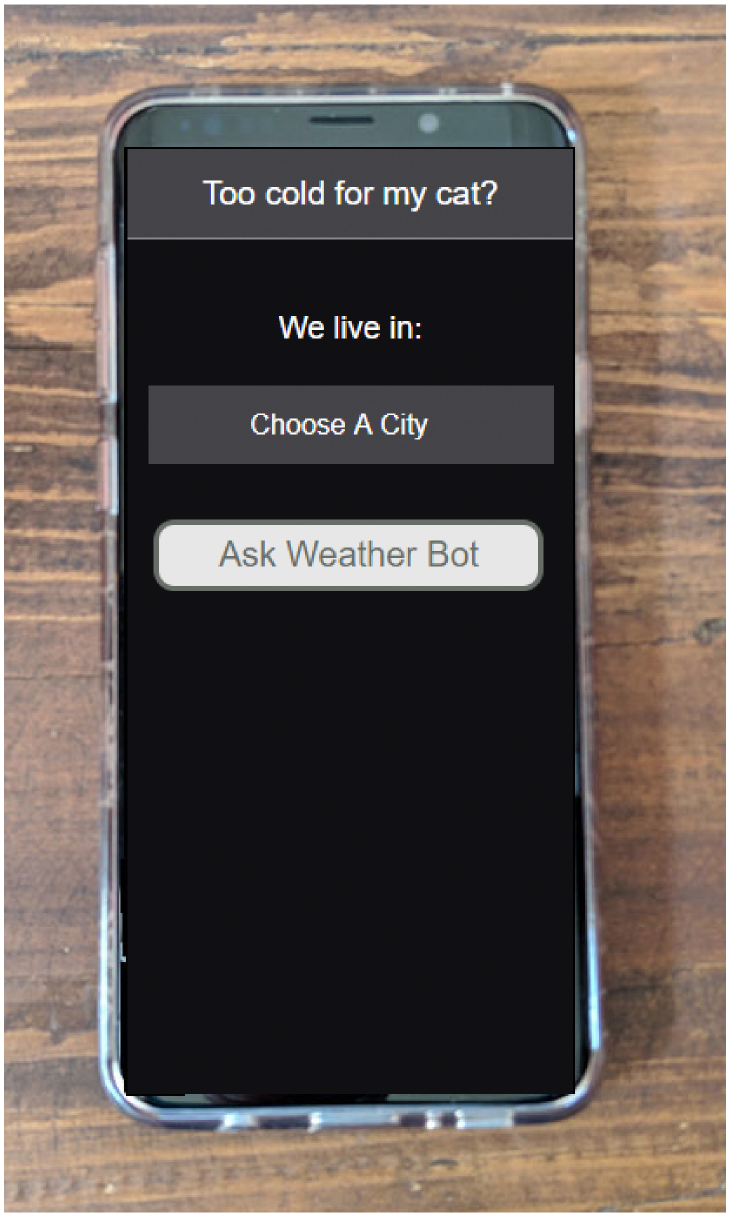
Metadata is a set of name-value pairs. You cannot modify object metadata after it is uploaded.

Header	Value
Cache-Control	max-age=0

- Click **Next**.
- **Double check it ;)**
- Click **Upload**.
- Verify connectivity by browsing to your tab (website) where you had the 404 error.

You should see your shiny new text based static web based application (see image below).

⚠️ PLEASE USE OR SWITCH TO CHROME for this, as later on we use **chrome only** features for the application.



- Press the section that says "Choose a city" it will give you a choice of cities, then press **Ask Weather Bot**.

As of right now, every city you choose will return **20 degrees**. This is because this is just a static website. Fear not we will add backend intelligence to this later on.

3. Steps to make local changes and upload your adjustments.

We want to make sure that when we make changes to local files that they are propagated correctly. So please make the following change to your local copy of `text.html`

From

```
<h1>Too cold for my cat?</h1>
```

To

```
<h1>Too hot for my cat?</h1>
```

- Upload the edited `text.html` to the S3 bucket:
  - Click **Upload**.
  - Click **Add files**.
  - Browse to the location of **text.html** on your local machine or drag and drop like before.
  - Click **Next**. Leave the permissions alone and click **Next**.
  - Again scroll down to **Metadata** and set the **Header** to `Cache-control` and set the **Value** to `max-age=0`.
  - **DOUBLE CHECK IT**.
  - Click **Save** and click **Next**.
  - Finally click **Upload**.
- Refresh your website

You should now see the changes. Awesome!

That's another one off our goal checklists done, let's see our progress.

Exercise goal checklist

1. ~~Create a simple chatbot using the lex console.~~
2. ~~Upload our website to S3.~~
3. Create a content delivery network and lock down S3.
4. Build an API gateway mock with CORS.
5. Build a Lambda mock, use IAM, push logs to CloudWatch.
6. Create and seed a database with weather data.
7. Enhance the lambda, so it can query the database.
8. Play with your new text based data driven application.
9. Create a LEX proxy using Lamba.
10. Enhance API gateway to use the LEX proxy.
11. Play with your new voice web application.

Well done. Please head back to the video content so we can discuss what to do next.