

CV Documentation - Part 2: Creation and deployment.

The first thing you have to do is make sure you have angular installed on your computer, if not, please read the angular official site and guide: <https://angular.io/guide/setup-local>

Having angular already installed, open the command prompt and go to the folder you want to create your angular project and run the next command:

```
ng new [App name]
```

After it finished, you'll have your own angular project to work with.

The next step is to create the components I listed above, for that run each one of the commands I'll list one by one:

```
cd [App name]
```

```
ng g c home
```

```
ng g c play
```

```
ng g c about
```

```
ng g c background
```

```
ng g c contact
```

```
ng g c menu
```

```
ng serve --open
```

In case you did everything right, you should now be on your browser on the default angular landing.

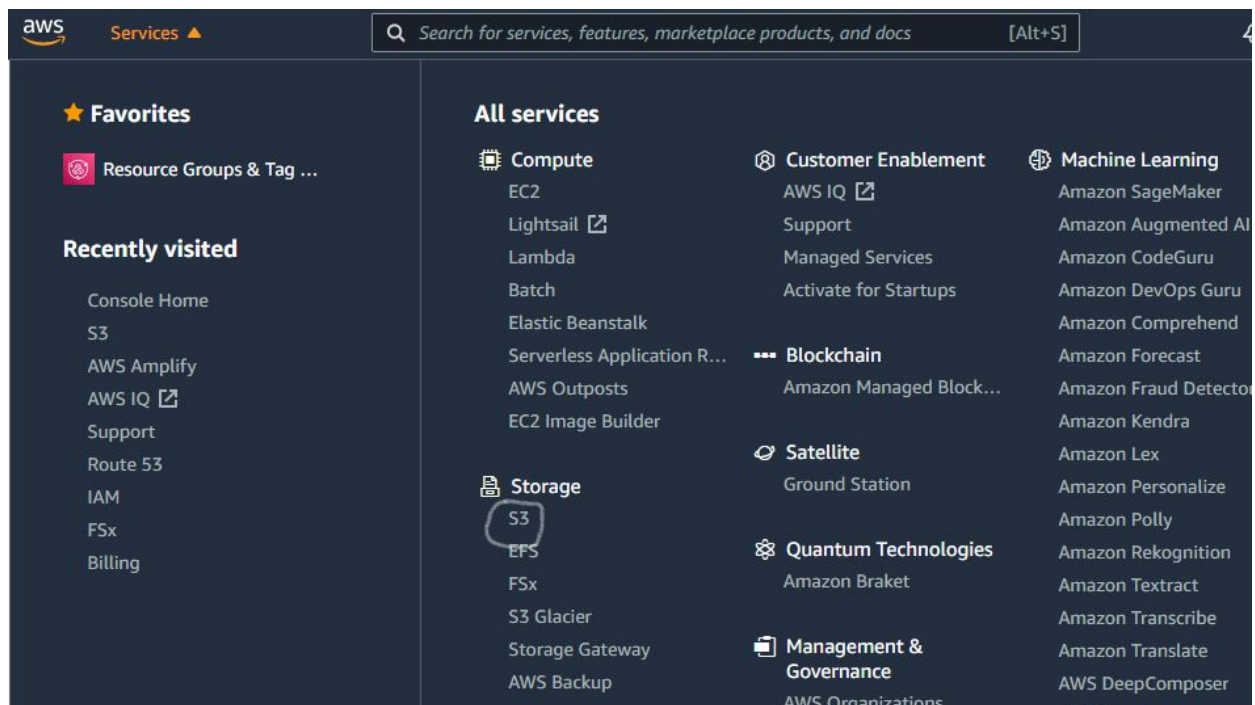
Congrats! You have now the base to make an awesome angular app.

The next step is to upload your app to an AWS storage, where we are going to be able to see our app online from our local files.

In this version I'll only be explaining how to upload the local files, be tune for the next version of this guide where I'll be explaining how to use a git repository to publish your app.

Now, for the deployment we need to create a new account on <https://aws.amazon.com/console/>. AWS offers a lot of free services, you can check it out. In our case, we need S3.

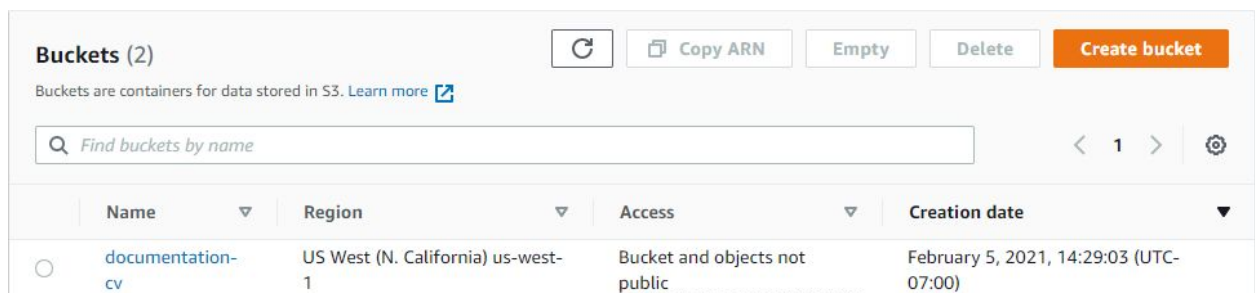
Once you get past the registration, on the services' navigation menu, search for the S3 title, under it, you'll find "Storage option".



The first thing we need to do is to create a bucket name but be careful, this name needs to be unique and chosen wisely as it will appear on your website URL

The region selection is pretty simple, just select the closest region to you, so the server will be able to load the website faster.

Click on "Create Bucket" at the bottom and that's it, you now have your S3 bucket. You should see something like this on your S3 dashboard:



If everything is correct, try going to your bucket just to check everything is alright, the URL will be:

(your bucket name).s3-website.(time zone name).amazonaws.com

You can find your time zone name on the bucket list, on the region description. In my case, it should be:

documentation-cv.s3-website.us-west-1.amazonaws.com

Assuming that everything is correct, you should be now seen a website like this:

403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: 04B3474745E65194
- HostId: BRwaCeb/1YPB7inlk2brQNgy7+Bym5tU1ekSCL2jyIDzDe5xr2gr3xyuCD1al/WN1Pxxm+BFPk=

An Error Occurred While Attempting to Retrieve a Custom Error Document

- Code: AccessDenied
 - Message: Access Denied
-

Congrats! You have created your bucket correctly. This error is due to the fact that we do not have access to the bucket



Now, to configure our bucket to be accessible, follow the next guide:



Open your bucket by clicking on its name, go to the “Properties” tab, go to the button and click on the “Edit” button in the “Static Web Hosting” section. Inside it, enable the static web hosting and leave everything else like this:

Static website hosting

- ☐ Disable
☒ Enable

Hosting type

- ☒ Host a static website
Use the bucket endpoint as the web address. [Learn more](#) 
- ☐ Redirect requests for an object
Redirect requests to another bucket or domain. [Learn more](#) 

 For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see [Using Amazon S3 Block Public Access](#) 

Index document


Specify the home or default page of the website.

Error document - *optional*

This is returned when an error occurs.

Go to the “Permissions” tab and press the “Edit” button in the “Block public access” section. Uncheck the “Block all public access” and just leave the first and second box checked:

Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#) 

- ☐ **Block *all* public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.
- ☒ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☒ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.
- ☐ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☐ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Save changes and go to the “Bucket Policy” section and click on “Edit”. Here we need to grant anonymous users access to our data. For that, we need to paste the next policy, just change your bucket name where is required:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AddPermission",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::(bucket name)/*"
    }
  ]
}
```

That's it, everything is ready for the deployment. The last step is to upload our angular app; make sure to build the app and upload the output as explained here:

Open your command prompt and go to the folder of your project. There, run this command:

```
ng build
```

You'll find a new folder on your project called "dist". This is the output of the angular app building. In case you don't know where the dist folder is, go to the "angular.json":

```
"architect": {
  "build": {
    "builder": "@angular-devkit/build-angular:browser",
    "options": {
      "outputPath": "src/dist/documentation-cv",
```


Now, you can just upload manually the folder to your bucket and that's it, the problem is that you'll need to upload the folder everything you make changes on your app, to fix that, I'll explain how to automatically upload your app.


Go to <https://console.aws.amazon.com/iam/home#/users>. Here we are going to create an user to give it access to the bucket.


Press on "Add User". Give a name to the user and select the "Programmatic access" in "Access type" and click "Next"

On permission, we are going to give our user the default administrator access. Select "Attach existing policies directly" then, check the "AdministratorAccess-Amplify" box.

▼ Set permissions









 Add user to group

 Copy permissions from existing user

 Attach existing policies directly

Create policy

Showing 641 results


	Policy name ▼	Type	Used as
<input type="checkbox"/>	 AdministratorAccess	Job function	Permissions policy (1)
<input checked="" type="checkbox"/>	 AdministratorAccess-Amplify	AWS managed	None
<input type="checkbox"/>	 AdministratorAccess-AWSElasticBeanstalk	AWS managed	None
<input type="checkbox"/>	 AlexaForBusinessDeviceSetup	AWS managed	None
<input type="checkbox"/>	 AlexaForBusinessFullAccess	AWS managed	None
<input type="checkbox"/>	 AlexaForBusinessGatewayExecution	AWS managed	None
<input type="checkbox"/>	 AlexaForBusinessLifesizeDelegatedAccessPolicy	AWS managed	None
<input type="checkbox"/>	 AlexaForBusinessPolyDelegatedAccessPolicy	AWS managed	None

Cancel Previous Next: Tags



Click “Next” until the user is created, and you get to this screen:

Add user

1 2 3 4 5

 **Success**
You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.
Users with AWS Management Console access can sign-in at: <https://722986132072.signin.aws.amazon.com/console>

Download .csv

	User	Access key ID	Secret access key
	documentation-cv	AKIA2QVKU4JUHLKKELJ 	***** Show

Close

I suggest downloading the .csv file, we are going to need the access key and the secret access key.

We need to install AWS CLI to be able to use AWS command on our command prompt. In case you have a Mac, you only need to run this command:

```
brew install awscli
```

In case you have Windows, please follow the AWS installation guide (just install the .msi file):
<https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-windows.html>

To check if you successfully installed AWS CLI, in the terminal run:

```
aws --version
```

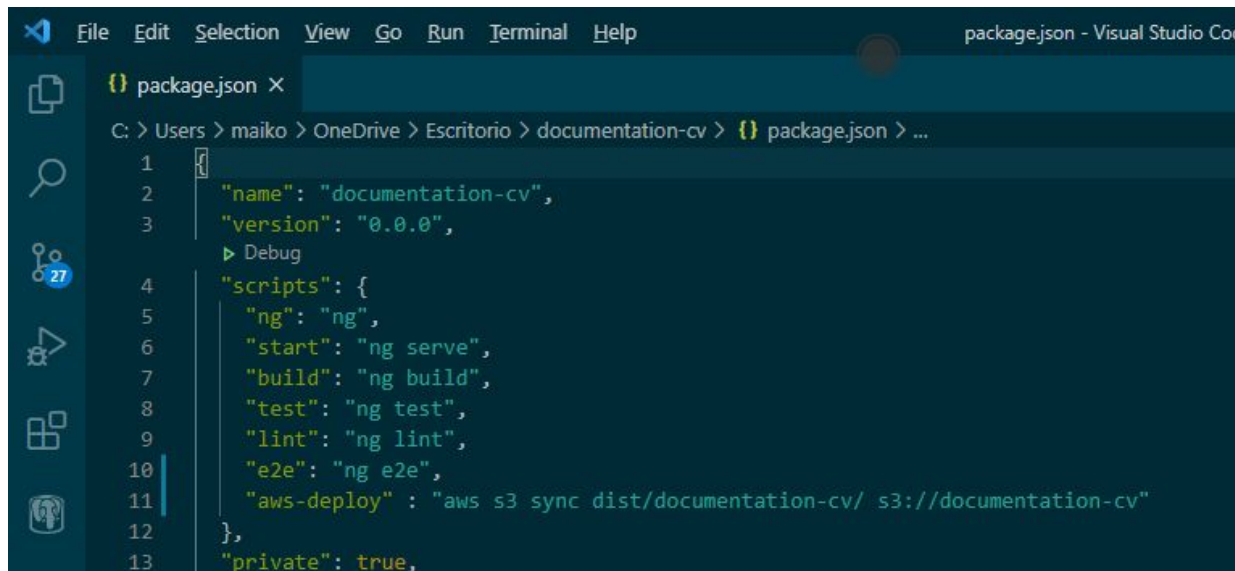
When you get sure you have AWS CLI, run:

```
aws configure
```

Enter your AWS Access Key ID, AWS Secret Access Key, Default region name (time zone name) and Default output format ("ENTER" will do).

On your project, go to the "package.json" and add this line:

```
"aws-deploy" : "aws s3 sync dist/(app name)/ s3://(bucket name)"
```



```
File Edit Selection View Go Run Terminal Help
package.json - Visual Studio Code
package.json x
C: > Users > maiko > OneDrive > Escritorio > documentation-cv > {} package.json > ...
1 {
2   "name": "documentation-cv",
3   "version": "0.0.0",
4   "scripts": {
5     "ng": "ng",
6     "start": "ng serve",
7     "build": "ng build",
8     "test": "ng test",
9     "lint": "ng lint",
10    "e2e": "ng e2e",
11    "aws-deploy" : "aws s3 sync dist/documentation-cv/ s3://documentation-cv"
12  },
13  "private": true,
```

And... that's it. Congrats! You made it, now everything you need to do everything you make any change on your app is run:

```
ng build && npm run aws-deploy
```


And go to your URL, right now, as we haven't done anything yet, you should see something like this:

