GOING SERVERLESS WITH YOUR OWN ONLINE TRANSLATOR

An easy AWS lambda use case

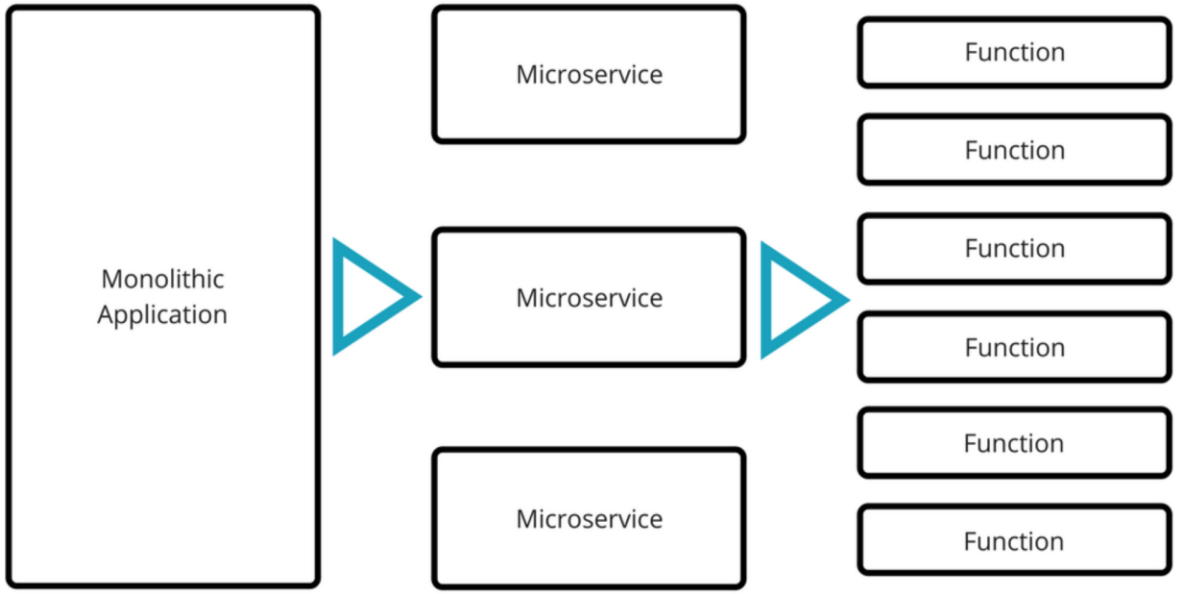
DEVWEEK endava

AGENDA

- INTRO TO SERVERLESS
- ARCHITECTURE USE CASES
- AWS SERVERLESS SERVICES
- BEST PRACTICES
- STEP BY STEP CONFIG
- TRANSLATOR DEMO
- TRANSLATOR BUDGET

INTRO TO SERVERLESS

Another name for serverless computing is Function-as-a-Service FaaS, referring to the way developers **assemble code into building blocks called functions.**



Following microservices approach where a monolith is split into small elements that can be scaled and updated separately and parallelly, **FaaS breaks each microservice even further.**

INTRO TO SERVERLESS

For example, for your home security camera you obviously don't want to record everything that happens on your street 24/7. That's why we use motion-activated cameras to detect suspicious behaviour when we're not at home.

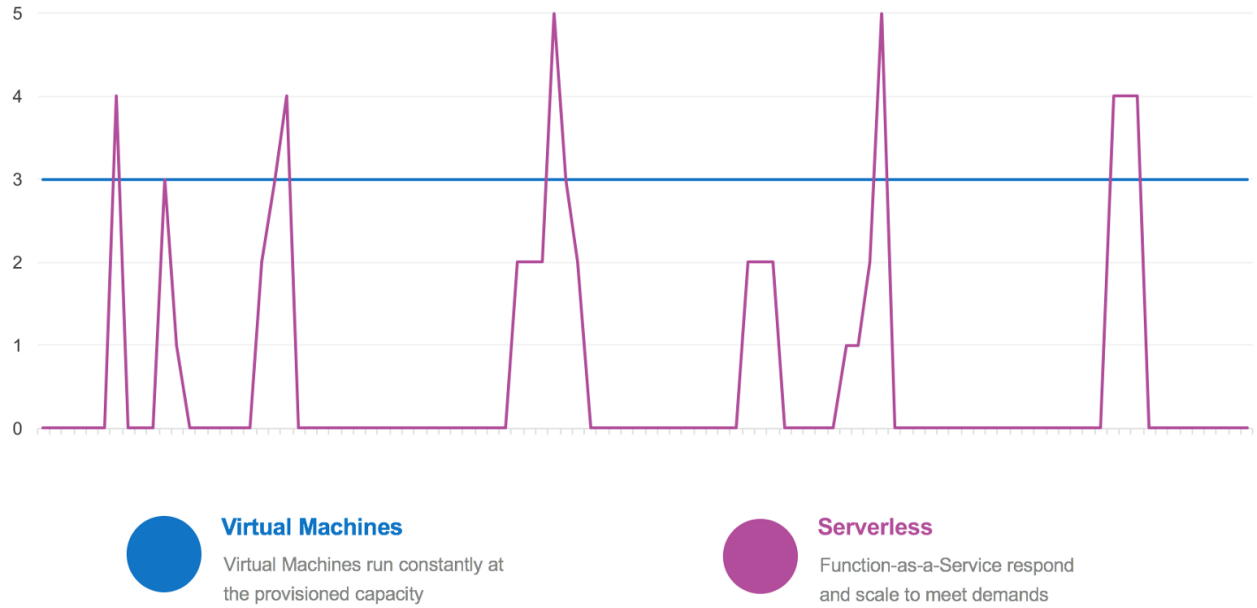
Serverless architecture works similarly, just like a motion sensor, it only works or run when **a pre-programmed event is triggered**.

Serverless is stateless, meaning it only executes a task and doesn't store or reuse requests.

INTRO TO SERVERLESS

Serverless approach is flexible and ideal for scaling applications. FaaS vendors takes each of your functions and runs them in containers. This allows you to scale them endlessly and automatically.

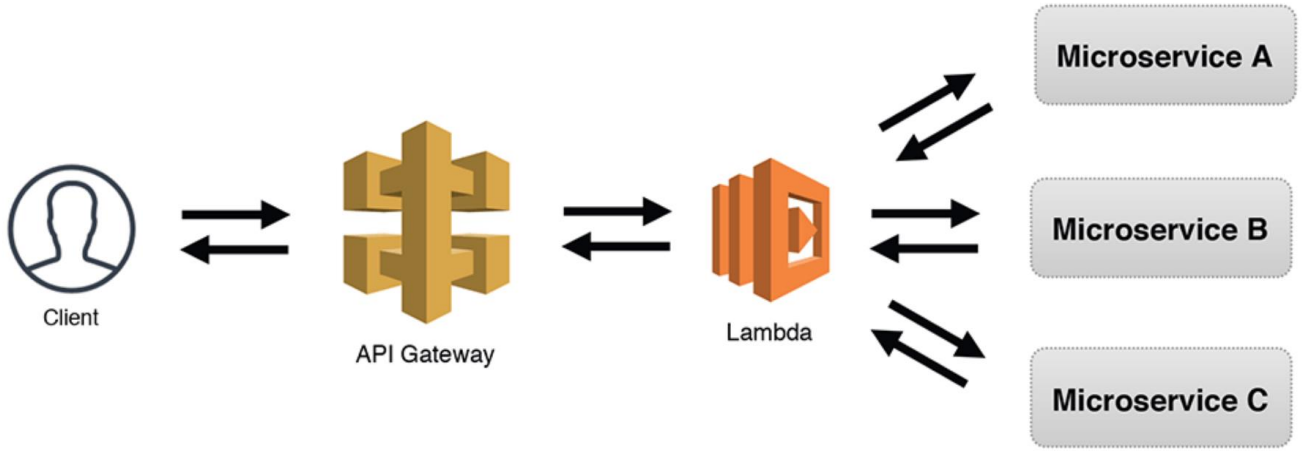
This is another difference between serverless and traditional cloud – here you don't have to purchase the assumed amount of resources; **you can be as flexible as possible.**



INTRO TO SERVERLESS

In a traditional cloud model, servers have to be ready to process requests at all times. Constant server availability leads to significant backend costs every month, irrespective of CPU time and memory that are actually used. It's like leaving your heater on all year long, regardless of the weather or change of seasons, and paying huge sums for unneeded electricity.

Alternatively, serverless vendors like AWS, allow you to **pay a fraction of a price per request**, which means that your costs will depend only on how much traffic you had this month.



INTRO TO SERVERLESS

Function-as-a-Service

Developers assemble code into building blocks called functions.

Event-driven coding

Function executes a task and doesn't store or reuse requests.

Scalable service

Small elements flexible and ideal for scale them automatically.

Billing per invocation

Costs will depend only on how much traffic your function had.

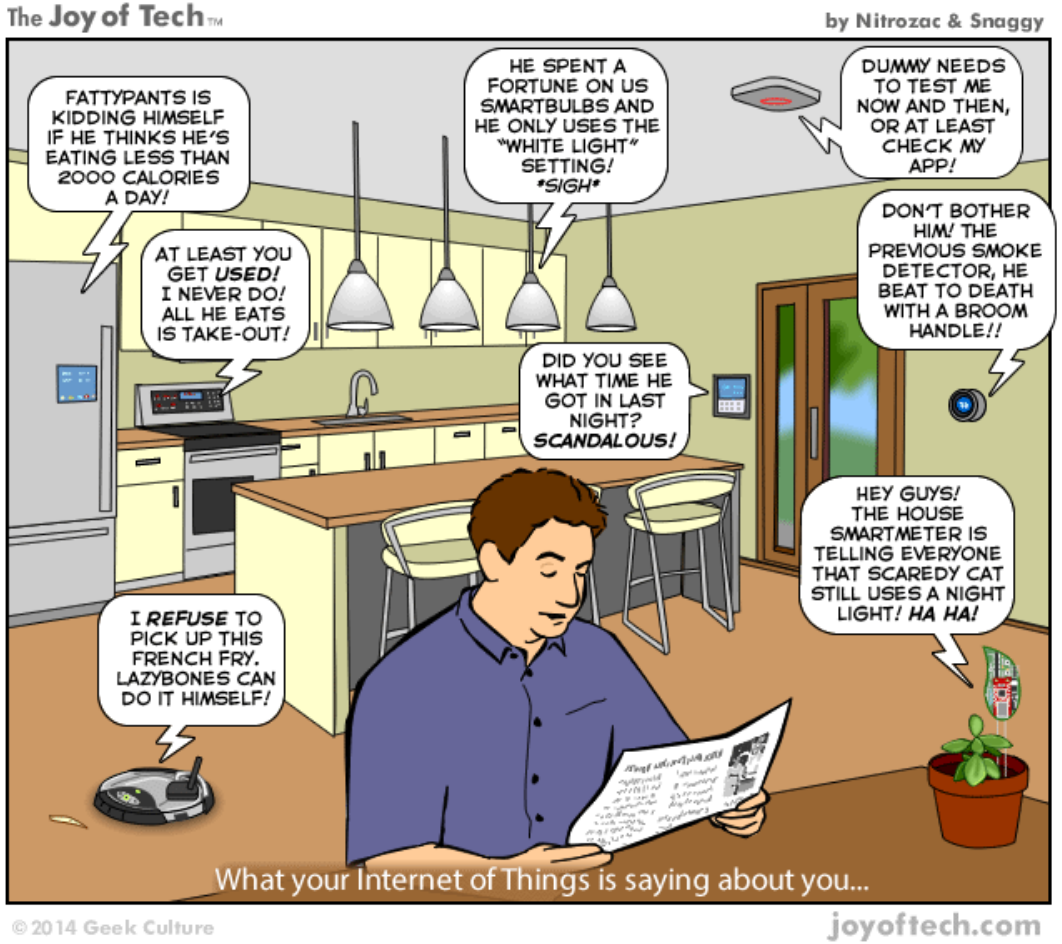
ARCHITECTURE USE CASES

Internet of Things applications

The real-time response nature of the serverless approach works great for IoT use cases.

Motion activated cameras, applications that react to changes in weather, temperature, or health conditions are perfect for the serverless

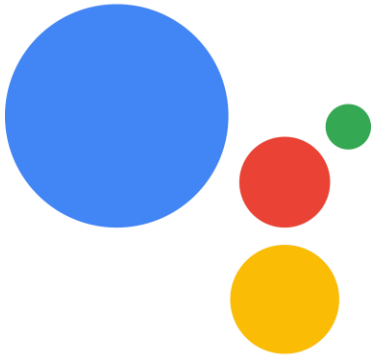
Your services won't be sitting idle 24/7.



ARCHITECTURE USE CASES

Virtual assistants and chatbots

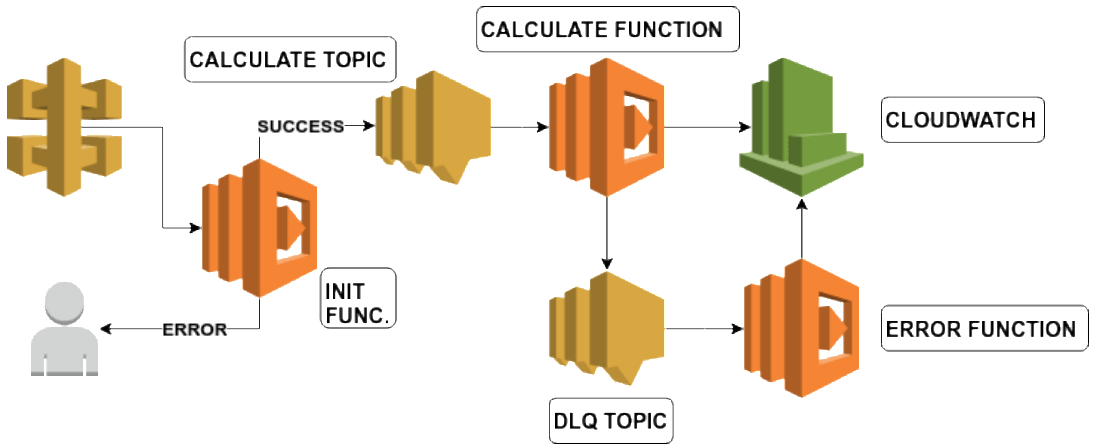
People using chats expect immediate responses which is why serverless data processing can be faster. As your application grows from one hundred to several thousand users, your processing time should also stay the same which is automated with FaaS.

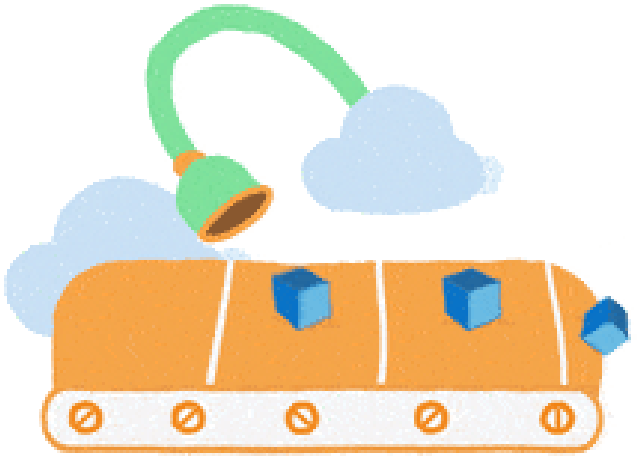


ARCHITECTURE USE CASES

Agile and Continuous Integration pipelines

The idea of running the code only when a certain event is triggered is perfectly in line with Agile or Continuous Integration principles. Separating the codebase into functions also helps with bug fixing and shipping updates. Serverless is an overall friendly way for maximum automation and rapid deployment processes.

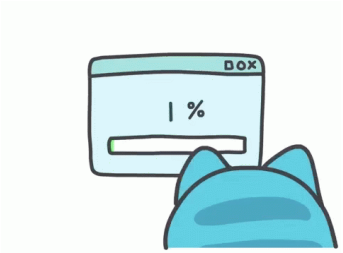




ARCHITECTURE USE CASES

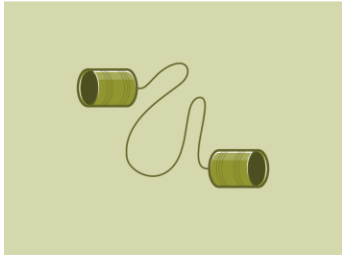
Sorting

Filtering



Real-time File processing

Real-time Stream processing



3rd-party API request

AWS SERVERLESS SERVICES

AWS provides **fully managed services** that can be used to build and run serverless applications.

This services handle application fault tolerance and availability capabilities. Tasks like provisioning, maintaining, and managing servers for backend components such as compute, databases, storage, message queueing, stream processing, and many more are no longer required.

AWS SERVERLESS SERVICES

AWS Lambda

Serverless computing system that lets you **automatically run code** in response to many types of events, such as HTTP requests from Amazon API gateway, table updates in Amazon DynamoDB, and state transitions.

The code runs on a high-availability computer infrastructure, where AWS performs all the administrative duties of that compute resource, like providing maintenance over the operating system, automatically scaling and managing capacity provisions, and handling security patching and code monitoring and logging.

https://aws.amazon.com/lambda/

AWS SERVERLESS SERVICES

AWS Cognito

Amazon Cognito lets you add user sign-up, sign-in, and access control to your web and mobile apps quickly and easily. Amazon Cognito scales to millions of users and supports sign-in with social identity providers, such as Facebook, Google, and Amazon, and enterprise identity providers via SAML 2.0.

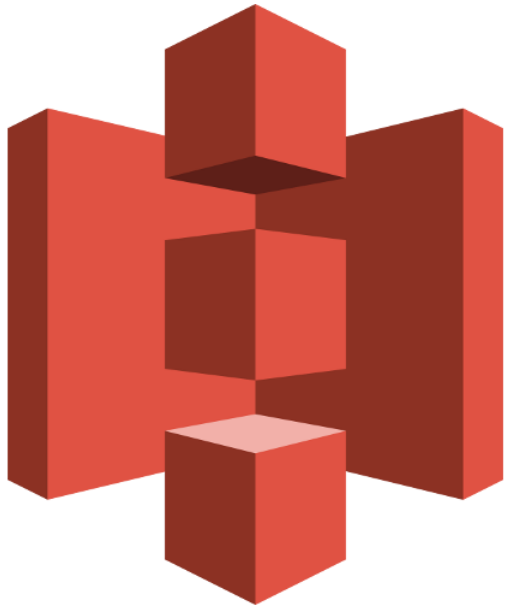
- Sign-In & Sign-up to mobile and web apps
- Federation (you like MS AD? Fine, fancy social networks? got it)
- Control AWS resources access, and restrict it to the public you want

https://aws.amazon.com/cognito/

AWS SERVERLESS SERVICES

AWS Simple Storage Service (S3)

Object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use it to store and protect any amount of data.



- Unlimited storage (Limit 5TB per file)
- Event notifications and Lifycicle management
- Host entire static websites
- Designed for 99.999999999% (11 9's) of data durability

https://aws.amazon.com/s3/

AWS SERVERLESS SERVICES

AWS CloudWatch

Monitoring and observability service that provides data and actionable insights to monitor your applications, respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health.

- Observability applications and insfraestucture
- Easily collect and store logs, metrics and events
- Built-in and custom metrics
- Log analytics and auto scalling

https://aws.amazon.com/cloudwatch/

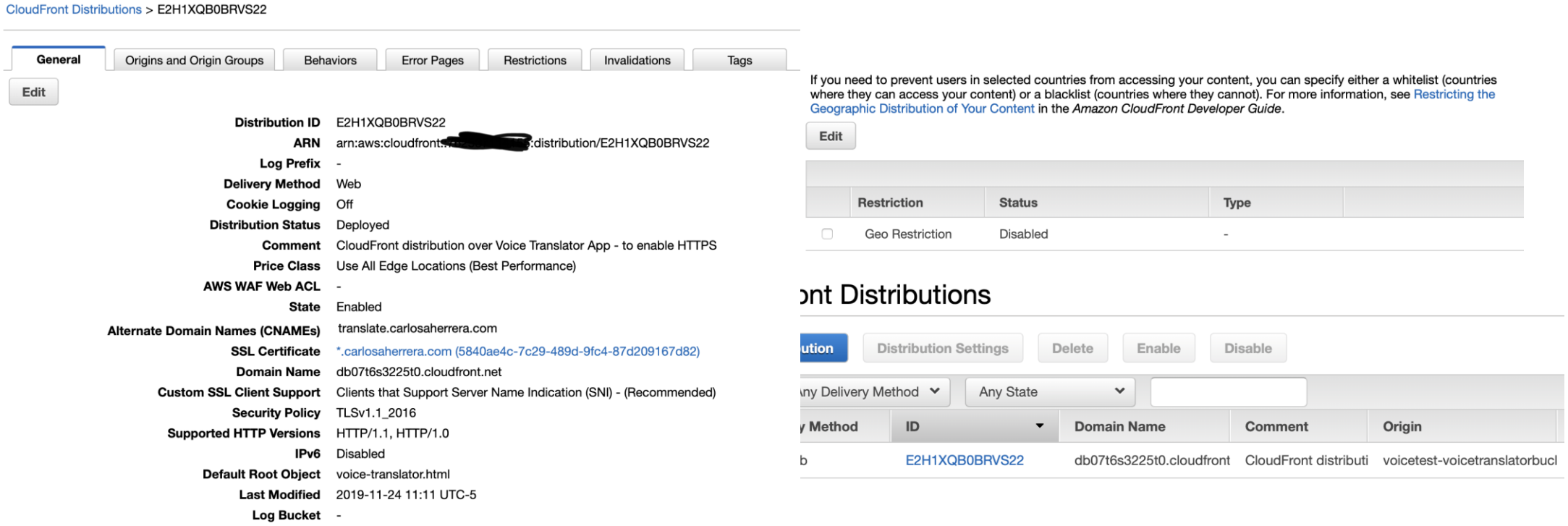
BEST PRACTICES

When you go serverles...

- Each function should do only one thing.
- Avoid using connection-based services e.g. RDBMS.
- Code for scaling and consider how it scales.
- Use as few libraries in your functions as possible.
- One function per route (if using HTTP).
- Learn to use messages and queues (async FTW).
- Data flows, not data lakes.
- Functions don’t call other functions.

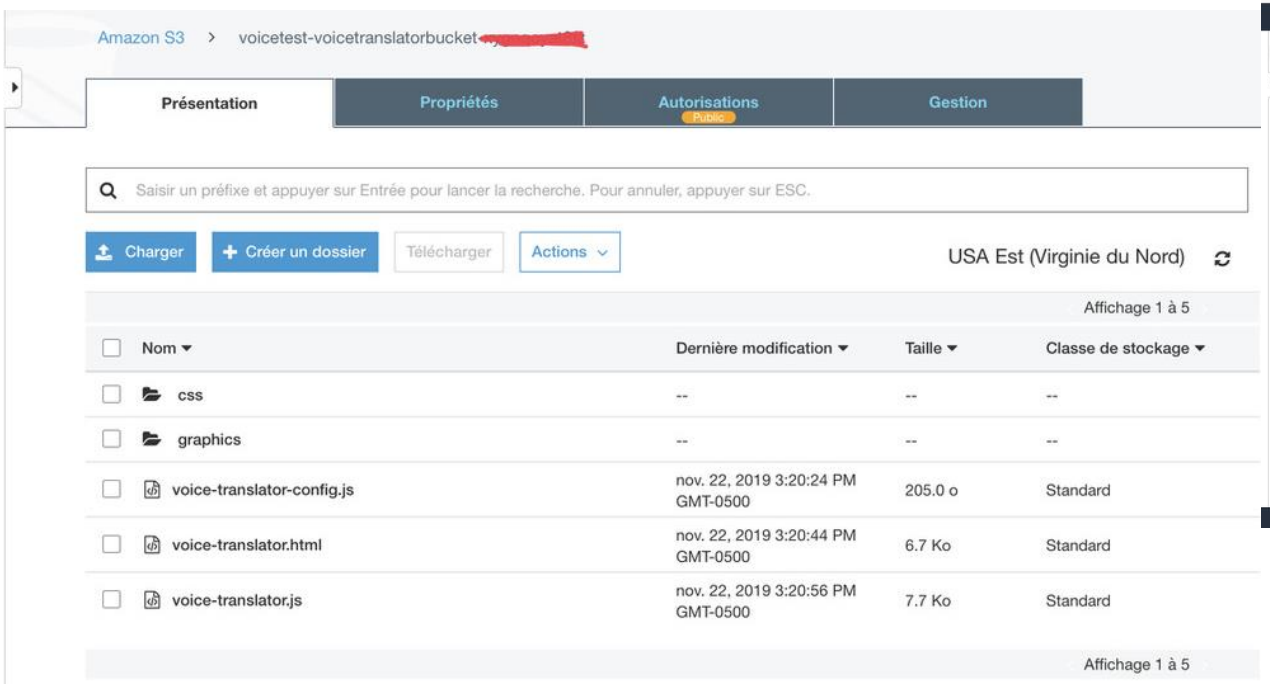
STEP BY STEP CONFIG

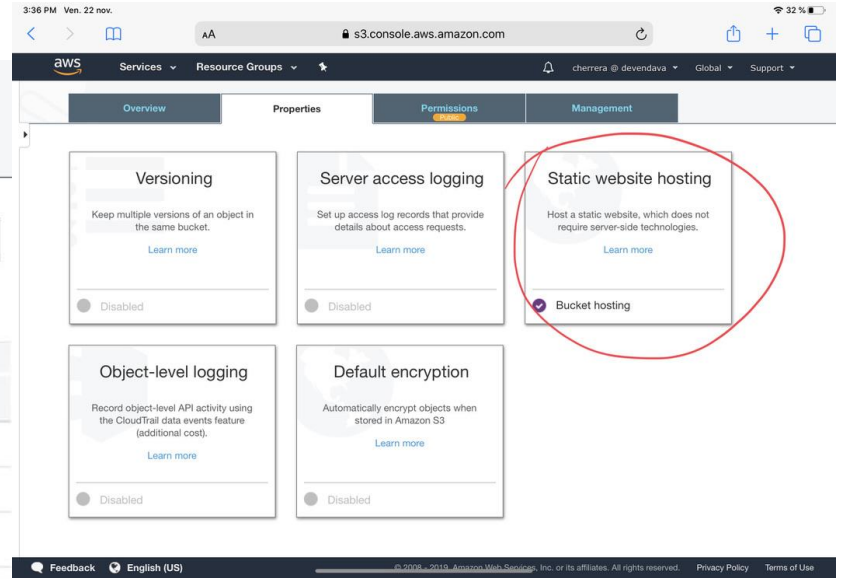
1. Allow access to your site with **Amazon CloudFront**, which allows you to get an HTTPS link to your page and which is required by some browsers to record audio.



STEP BY STEP CONFIG

2. Host your page on **Amazon S3**, which simplifies the whole solution. This is also the place to save the input audio file recorded in the browser.

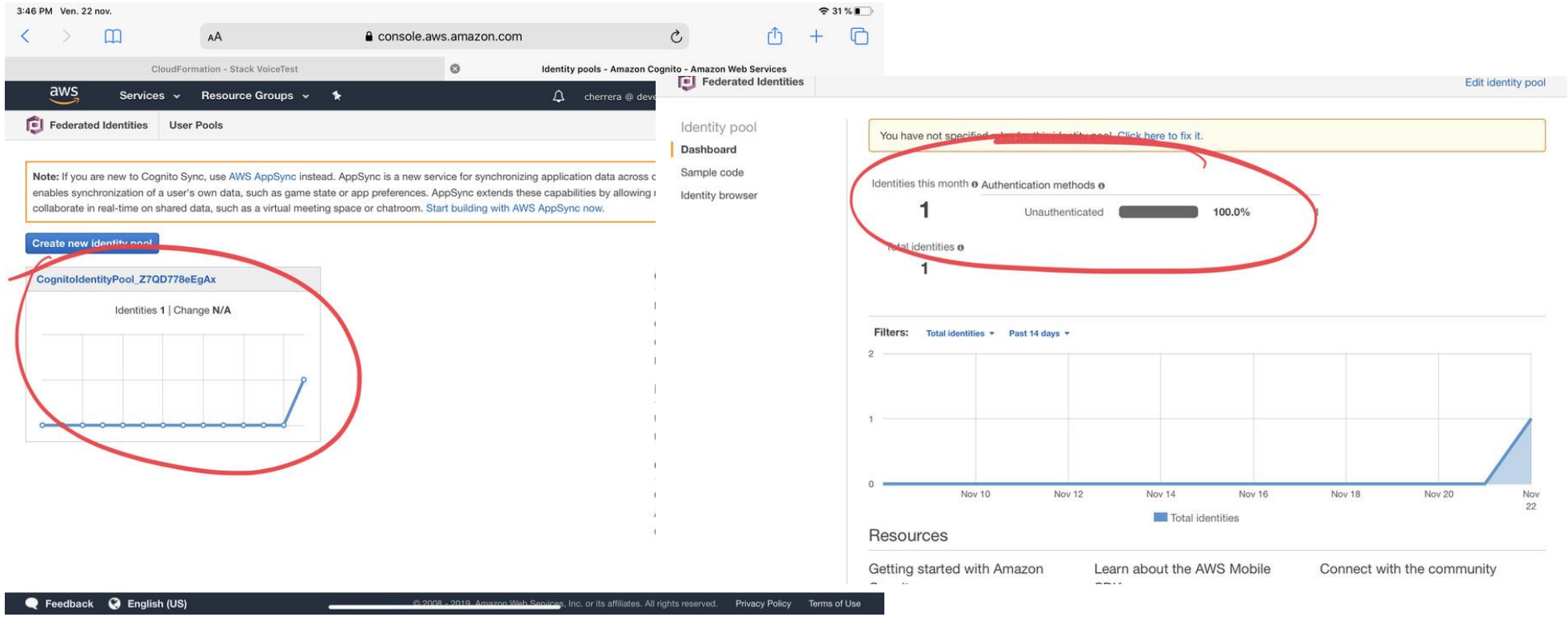




DEVWEEK endava

STEP BY STEP CONFIG

3. Gain secure access to S3 and Lambda from the browser with **Amazon Cognito**.



STEP BY STEP CONFIG

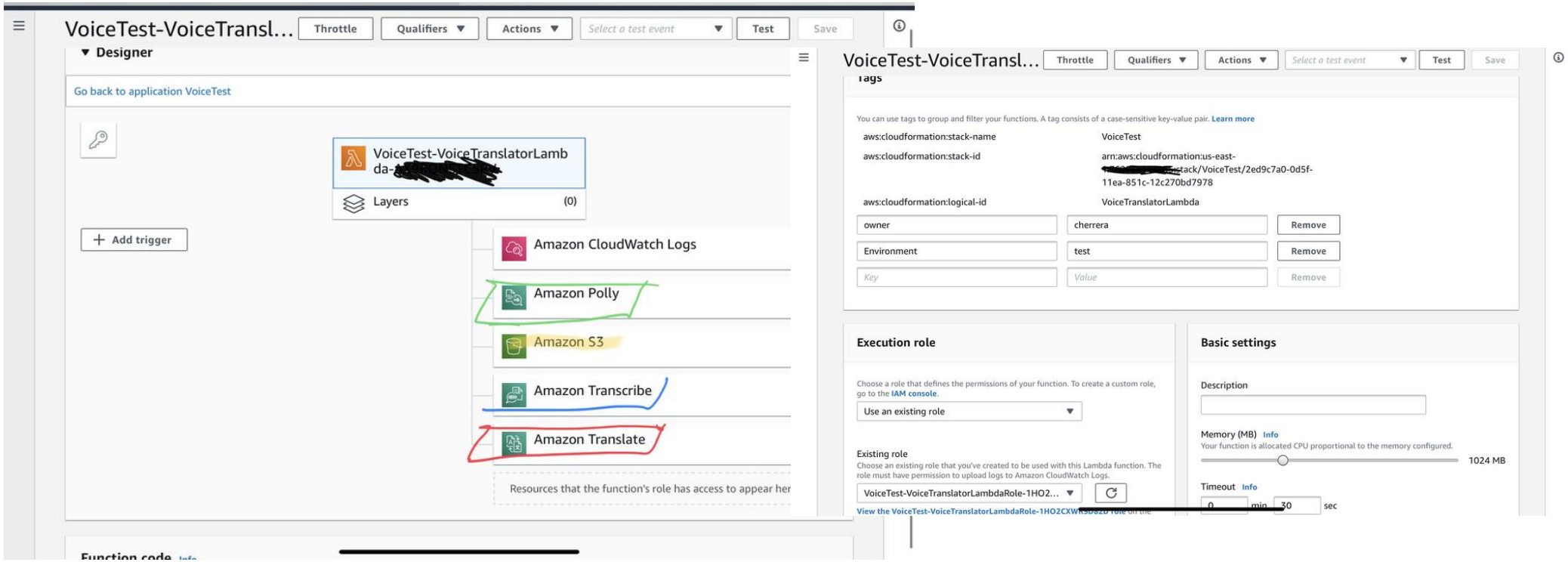
```
AWS.config.update({
  region: awsRegion,
  credentials: new AWS.CognitoIdentityCredentials({
    IdentityPoolId: IdentityPoolId
  })
});
```

```
CognitoUnAuthorizedRole:
  Type: "AWS::IAM::Role"
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
          Principal:
            Federated: "cognito-identity.amazonaws.com"
          Action:
            - "sts:AssumeRoleWithWebIdentity"
    Policies:
      - PolicyName: "CognitoUnauthorizedPolicy"
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
            - Effect: "Allow"
              Action:
                - "lambda:InvokeFunction"
              Resource:
                - !GetAtt VoiceTranslatorLambda.Arn
              Effect: Allow
              Action:
                - "s3:PutObject"
              Resource:
                Fn::Join:
                  - ""
                  - - "arn:aws:s3:::"
                    - Ref: "VoiceTranslatorBucket"
                    - "/*"
```

```
1  var bucketName = "voicetest-voicetranslatorbucket-
2  var IdentityPoolId = "us-east-1:d55044d7-f714-40cf-aebe-9723cb
3  var lambdaFunction = "VoiceTest-VoiceTranslatorLambda-AX9RQN
4
```

STEP BY STEP CONFIG

4. Save the input audio file on S3 and invoke a **Lambda function**. In the input of the function, provide the name of audio file, and pass the source and target language.



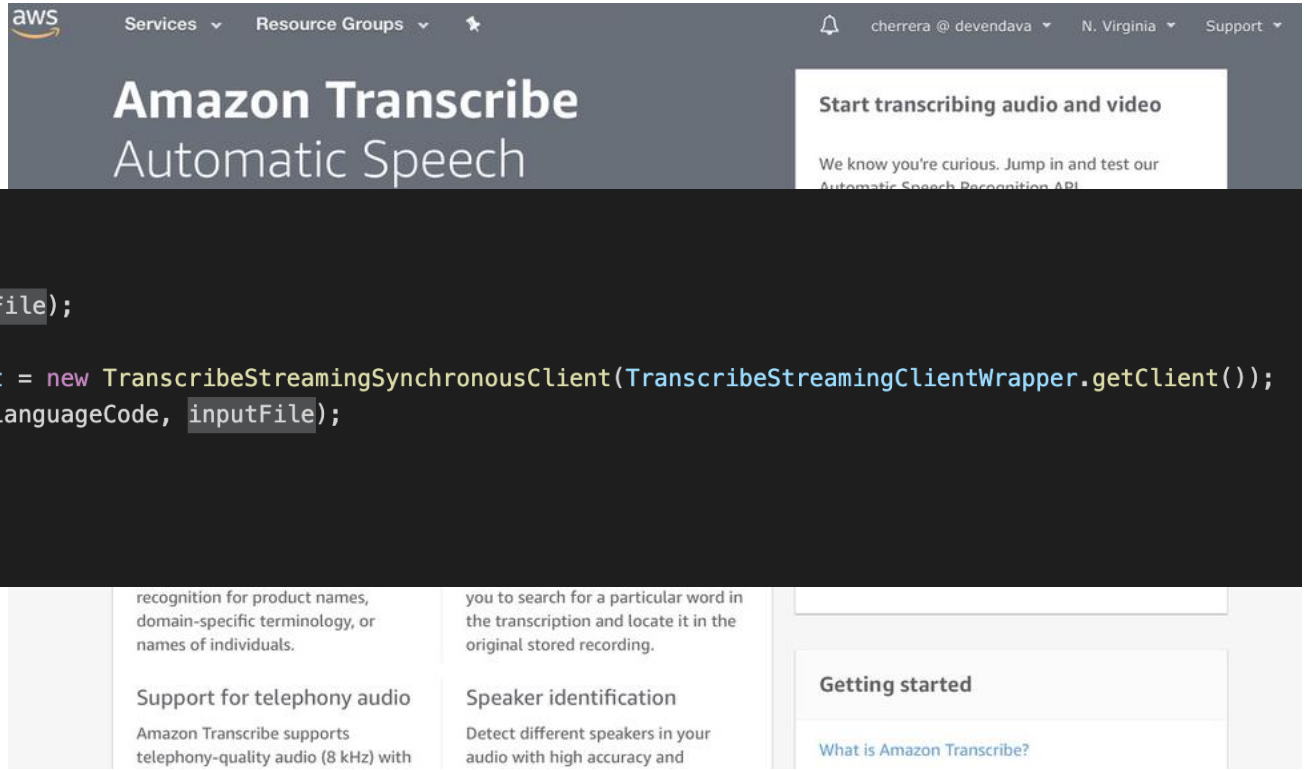
STEP BY STEP CONFIG

4. Save the input audio file on S3 and invoke a **Lambda function**. In the input of the function, provide the name of audio file, and pass the source and target language.

```
s3.upload(
  {
    Key: inputKey,
    Body: blob
  },
  function(err, data) {
    if (err) {
      return alert("There was an error uploading your recording: ", err.message);
    } else {
      var lambda = new AWS.Lambda({ region: awsRegion, apiVersion: "2015-03-31" });
      var input = {
        FunctionName: lambdaFunction,
        InvocationType: "RequestResponse",
        LogType: "None",
        Payload: JSON.stringify({
          bucket: bucketName,
          key: inputKey,
          sourceLanguage: source_language,
          targetLanguage: target_language
        })
      };

      lambda.invoke(input, function(err, data) {
        if (err) {
          console.log(err);
          alert("There was a problem with Lambda function!!! ");
        } else {
          var resultUrl = data.Payload.replace(/['"]+/g, "");
          resetView();
          document.getElementById("audio-output").innerHTML =
            '<audio controls autoplay><source src="' +
            resultUrl +
            '" type="audio/mpeg"></audio><br/>';
        }
      });
    }
  }
);
}
```

STEP BY STEP CONFIG

5. Convert audio into text with **Amazon Transcribe**.




```
File inputFile = new File("/tmp/input.wav");

s3.getObject(new GetObjectRequest(bucket, key), inputFile);

TranscribeStreamingSynchronousClient synchronousClient = new TranscribeStreamingSynchronousClient(TranscribeStreamingClientWrapper.getClient());
String transcript = synchronousClient.transcribeFile(languageCode, inputFile);

logger.log("Transcript: " + transcript);

return transcript;
```

STEP BY STEP CONFIG

6. Translate the transcribed text from one language to another with **Amazon Translate**.

```
TranslateTextRequest request = new TranslateTextRequest().withText(text)
        .withSourceLanguageCode(sourceLanguage)
        .withTargetLanguageCode(targetLanguage);
TranslateTextResult result  = translate.translateText(request);

String translatedText = result.getTranslatedText();

logger.log("Translation: " + translatedText);

return translatedText;
```

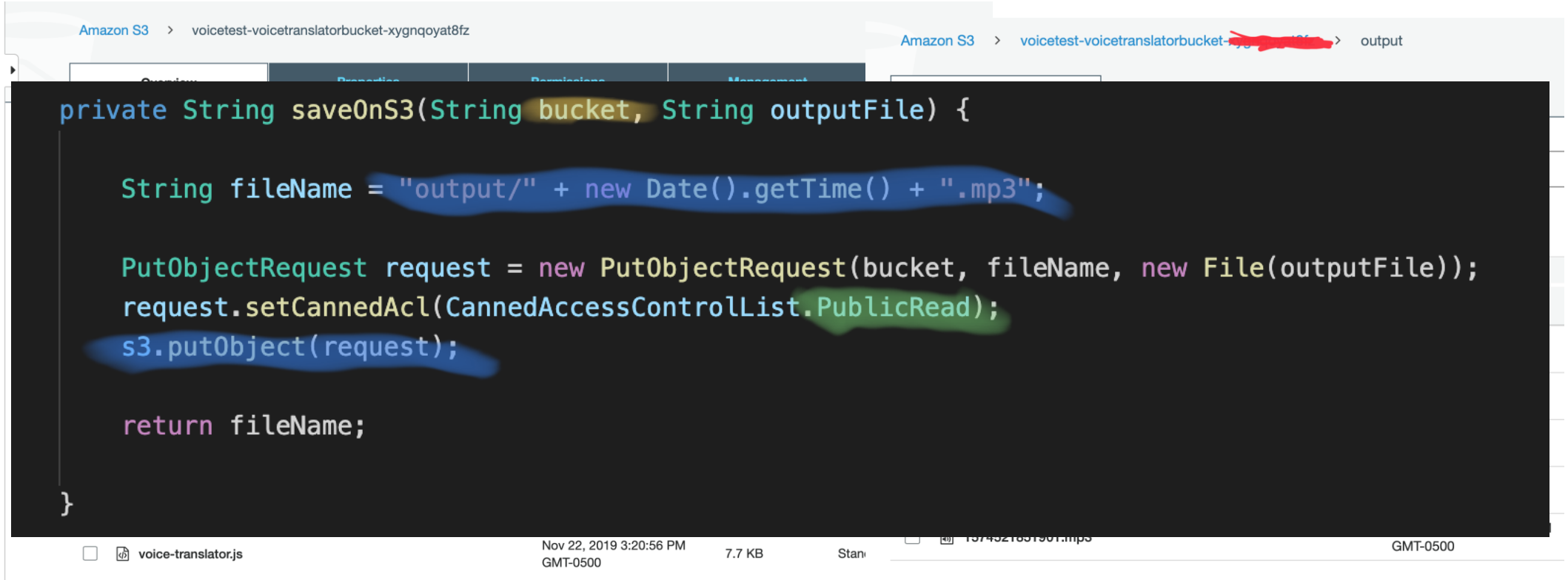
STEP BY STEP CONFIG

7.

```
String outputFileName = "/tmp/output.mp3";

SynthesizeSpeechRequest synthesizeSpeechRequest = new SynthesizeSpeechRequest()
        .withOutputFormat(OutputFormat.Mp3)
        .withVoiceId(voiceId)
        .withText(text);

try (FileOutputStream outputStream = new FileOutputStream(new File(outputFileName))) {
    SynthesizeSpeechResult synthesizeSpeechResult = polly.synthesizeSpeech(synthesizeSpeechRequest);
    byte[] buffer = new byte[2 * 1024];
    int readBytes;

    try (InputStream in = synthesizeSpeechResult.getAudioStream()){
        while ((readBytes = in.read(buffer)) > 0) {
            outputStream.write(buffer, 0, readBytes);
        }
    }

} catch (Exception e) {
    logger.log(e.toString());
}

return outputFileName;
```

STEP BY STEP CONFIG

8. Save the output audio file back to S3, and then return the file name to your page. You could return the audio file itself, but for simplicity, save it on S3 and just return its name.



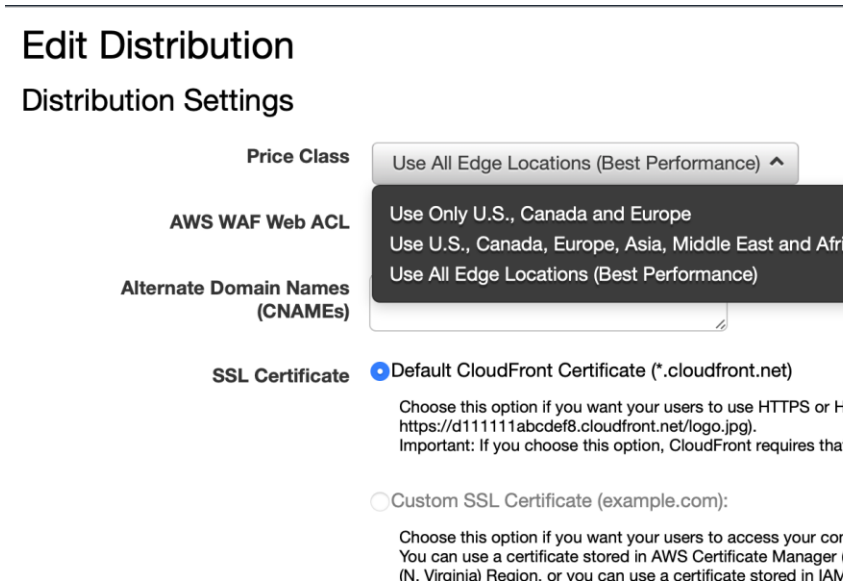
STEP BY STEP CONFIG

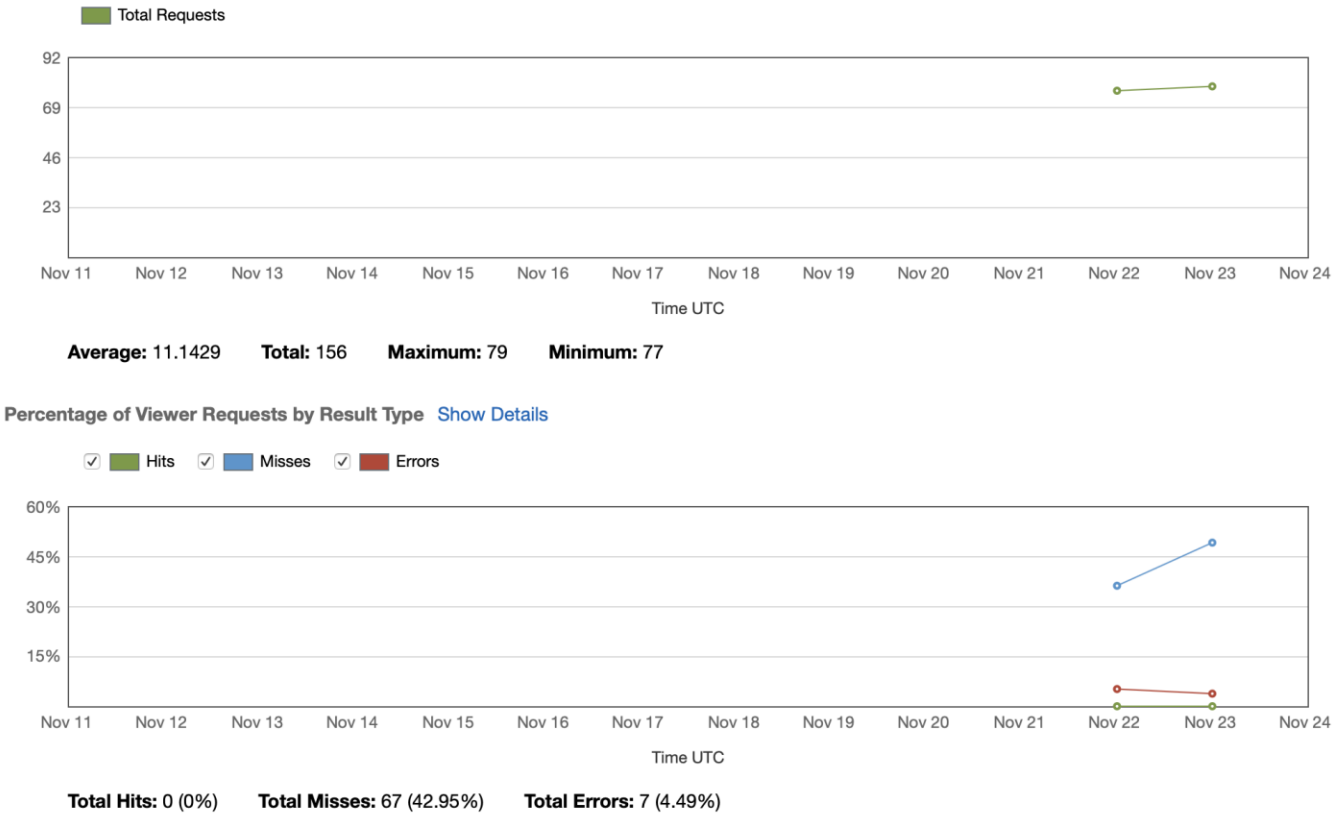
9. Automatically play the translated audio to the user.


```
} else {
  var resultUrl = data.Payload.replace(/['"]+/g, "");
  resetView();
  document.getElementById("audio-output").innerHTML =
    '<audio controls autoplay><source src="' +
    resultUrl +
    '" type="audio/mpeg"></audio><br/>';
}
```

STEP BY STEP CONFIG

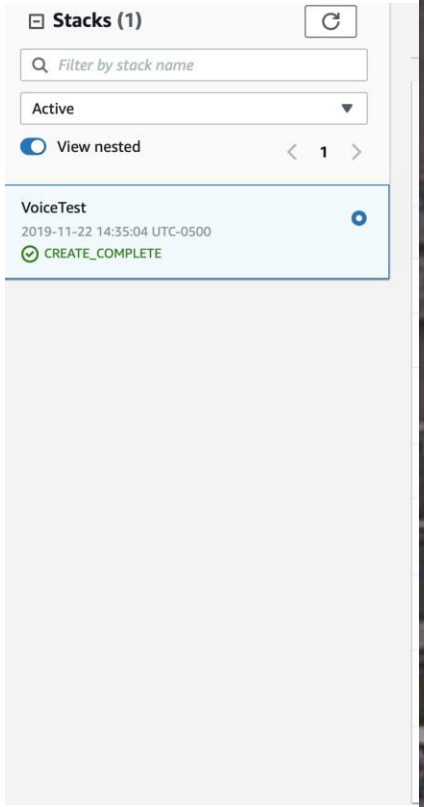
10. Accelerate the speed of delivering the file with **CloudFront**.

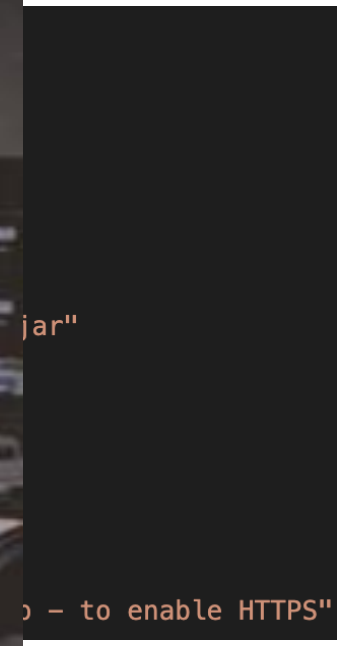




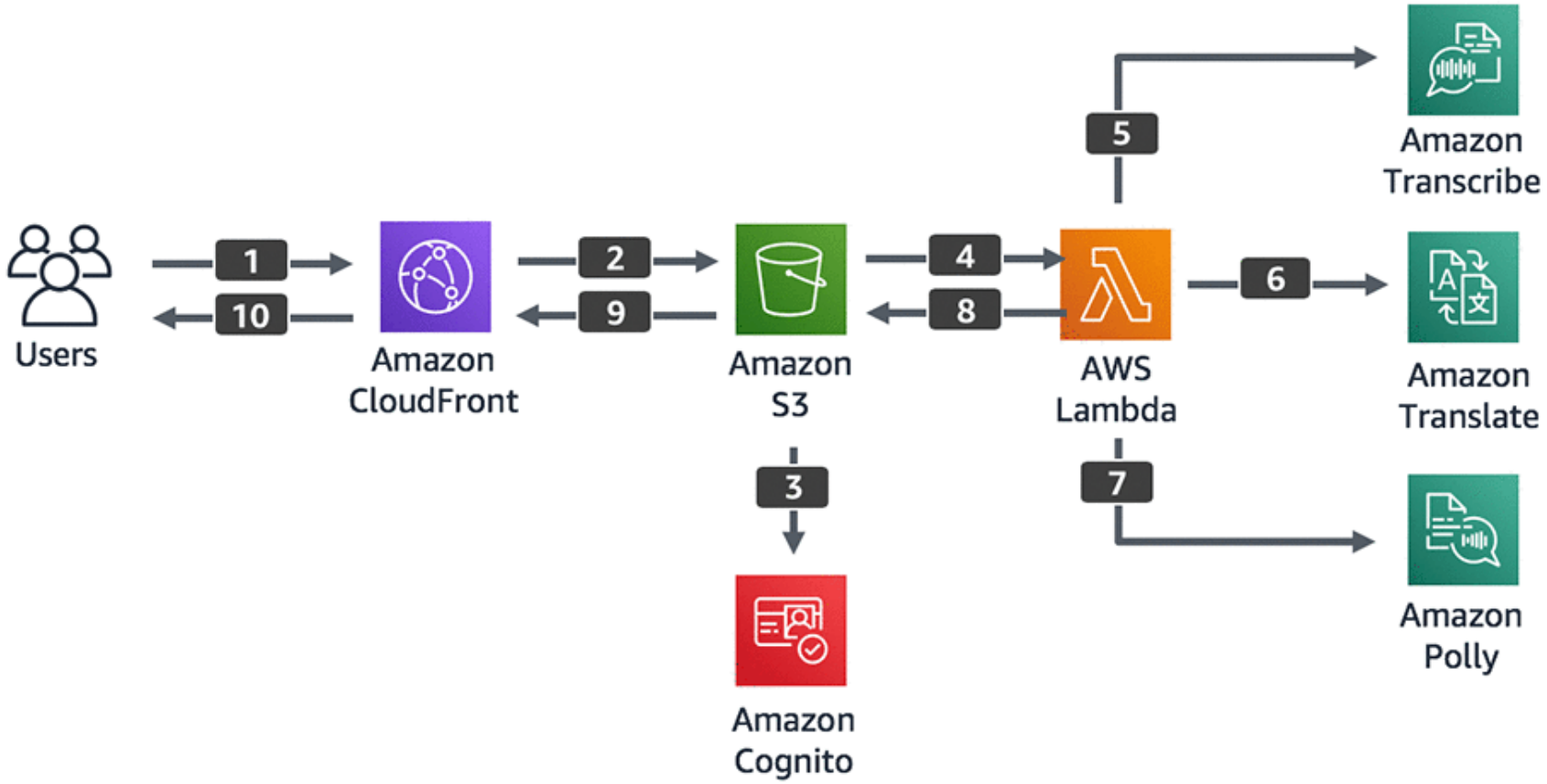
DOING THIS MANUALLY? … NONSENSE

CLOUDFORMATION/IaC TO THE RESCUE

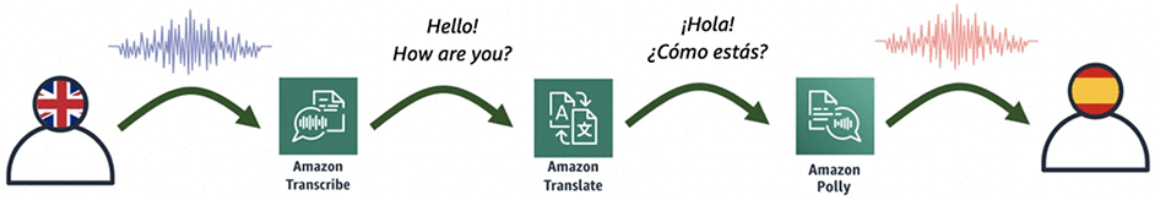




FLOW OF SERVICE INTERACTIONS



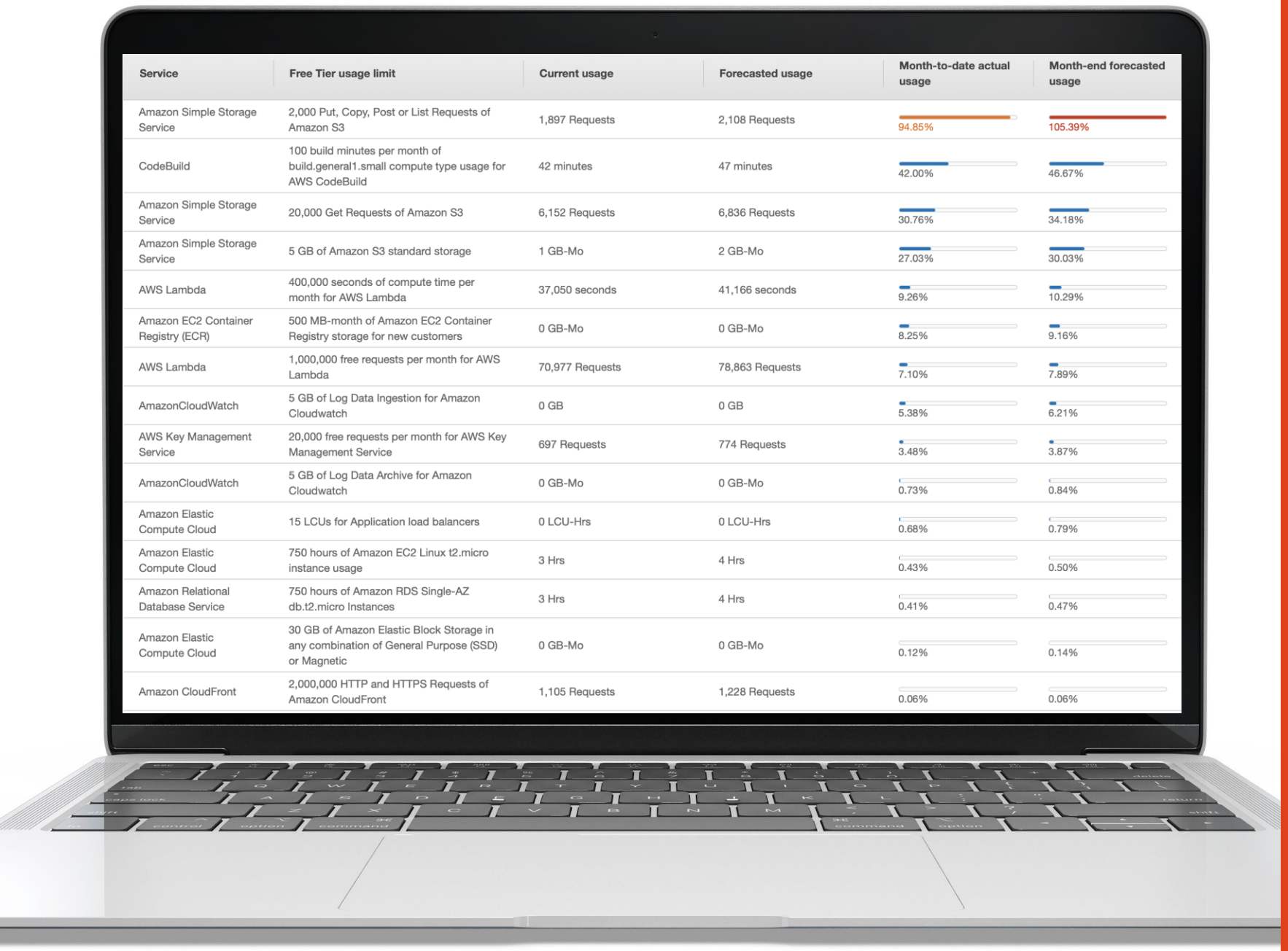
DEMO TIME!



https://translate.carlosaherrera.com

SERVERLESS TRANSLATOR DEMO

https://github.com/Carlos4ndresh/voicetranslateapp/issues



Service	Free Tier usage limit	Current usage	Forecasted usage	Month-to-date actual usage	Month-end forecasted usage
Amazon Simple Storage Service	2,000 Put, Copy, Post or List Requests of Amazon S3	1,897 Requests	2,108 Requests	94.85%	105.39%
CodeBuild	100 build minutes per month of build.general1.small compute type usage for AWS CodeBuild	42 minutes	47 minutes	42.00%	46.67%
Amazon Simple Storage Service	20,000 Get Requests of Amazon S3	6,152 Requests	6,836 Requests	30.76%	34.18%
Amazon Simple Storage Service	5 GB of Amazon S3 standard storage	1 GB-Mo	2 GB-Mo	27.03%	30.03%
AWS Lambda	400,000 seconds of compute time per month for AWS Lambda	37,050 seconds	41,166 seconds	9.26%	10.29%
Amazon EC2 Container Registry (ECR)	500 MB-month of Amazon EC2 Container Registry storage for new customers	0 GB-Mo	0 GB-Mo	8.25%	9.16%
AWS Lambda	1,000,000 free requests per month for AWS Lambda	70,977 Requests	78,863 Requests	7.10%	7.89%
AmazonCloudWatch	5 GB of Log Data Ingestion for Amazon Cloudwatch	0 GB	0 GB	5.38%	6.21%
AWS Key Management Service	20,000 free requests per month for AWS Key Management Service	697 Requests	774 Requests	3.48%	3.87%
AmazonCloudWatch	5 GB of Log Data Archive for Amazon Cloudwatch	0 GB-Mo	0 GB-Mo	0.73%	0.84%
Amazon Elastic Compute Cloud	15 LCUs for Application load balancers	0 LCU-Hrs	0 LCU-Hrs	0.68%	0.79%
Amazon Elastic Compute Cloud	750 hours of Amazon EC2 Linux t2.micro instance usage	3 Hrs	4 Hrs	0.43%	0.50%
Amazon Relational Database Service	750 hours of Amazon RDS Single-AZ db.t2.micro Instances	3 Hrs	4 Hrs	0.41%	0.47%
Amazon Elastic Compute Cloud	30 GB of Amazon Elastic Block Storage in any combination of General Purpose (SSD) or Magnetic	0 GB-Mo	0 GB-Mo	0.12%	0.14%
Amazon CloudFront	2,000,000 HTTP and HTTPS Requests of Amazon CloudFront	1,105 Requests	1,228 Requests	0.06%	0.06%

TRANSLATOR BUDGET

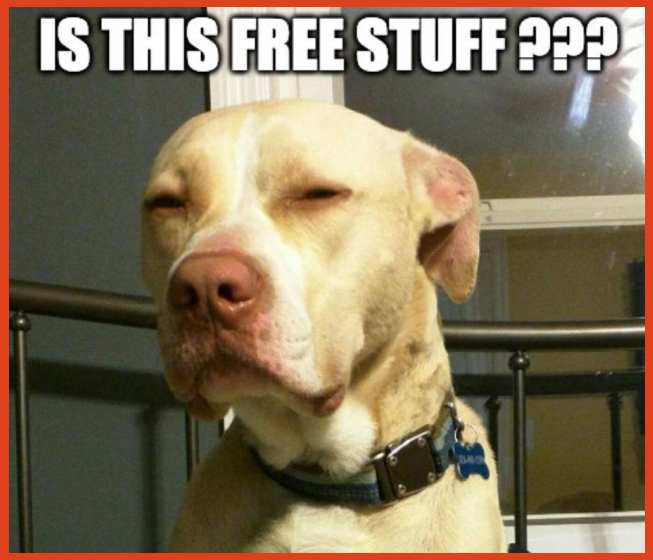
This demo used the following services:

- S3
- Lambda
- Cloudwatch
- KMS
- CloudFront
- Cognito

Costs, the not so cool side...

DEWEEK endava

CloudFront		**$0.00**
Europe		**$0.00**
Amazon CloudFront EU-Requests-Tier1		$0.00
$0.000 per request - HTTP or HTTPS under the global monthly free tier	1.000 Requests	$0.00
Amazon CloudFront EU-Requests-Tier2-HTTPS		$0.00
$0.000 per request - HTTP or HTTPS under the global monthly free tier	54.000 Requests	$0.00
Bandwidth		$0.00
$0.000 per GB - data transfer out under the global monthly free tier	0.000496 GB	$0.00
South America		**$0.00**
Amazon CloudFront SA-Requests-Tier1		$0.00
$0.000 per request - HTTP or HTTPS under the global monthly free tier	8.000 Requests	$0.00
Amazon CloudFront SA-Requests-Tier2-HTTPS		$0.00
$0.000 per request - HTTP or HTTPS under the global monthly free tier	744.000 Requests	$0.00
Bandwidth		$0.00
$0.000 per GB - data transfer out under the global monthly free tier	0.006 GB	$0.00
United States		**$0.00**
Amazon CloudFront US-Requests-Tier1		$0.00
$0.000 per request - HTTP or HTTPS under the global monthly free tier	5.000 Requests	$0.00
Amazon CloudFront US-Requests-Tier2-HTTPS		$0.00
$0.000 per request - HTTP or HTTPS under the global monthly free tier	293.000 Requests	$0.00
Bandwidth		$0.00
$0.000 per GB - data transfer out under the global monthly free tier	0.003 GB	$0.00
Lambda		**$0.00**
US East (N. Virginia)		**$0.00**
AWS Lambda Lambda-GB-Second		$0.00
AWS Lambda - Compute Free Tier - 400,000 GB-Seconds - US East (Northern Virginia)	407.700 seconds	$0.00
AWS Lambda Request		$0.00
AWS Lambda - Requests Free Tier - 1,000,000 Requests - US East (Northern Virginia)	60.000 Requests	$0.00
US West (Oregon)		**$0.00**



You can run this example paying USD $0 using

AWS FreeTier (12 months)

Check each service **FreeTier limit,** not everything that shines is gold!

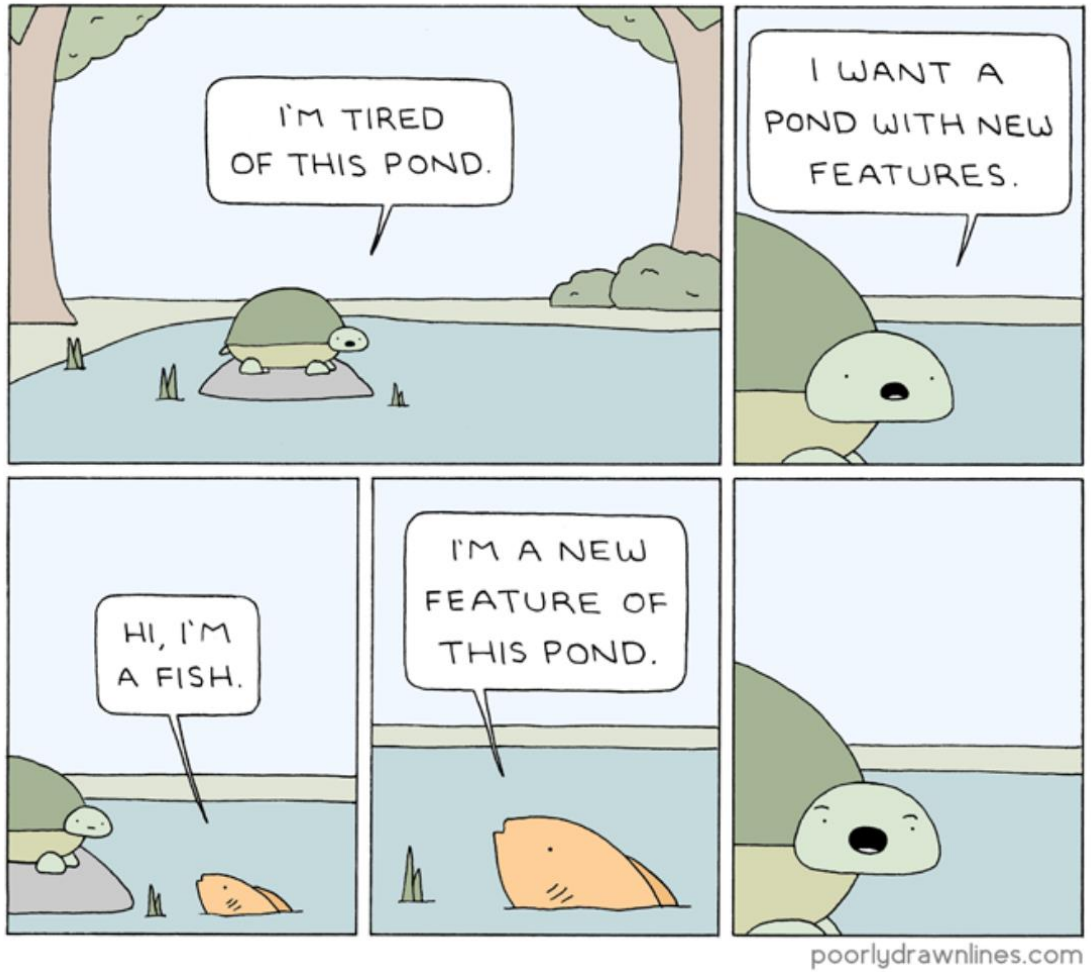
There are services that are not FreeTier included but, the trend it's that all will be in the future…

DEWEEK endava

Simple Storage Service		**$0.00**
US East (N. Virginia)		**$0.00**
Amazon Simple Storage Service Requests-Tier1		$0.00
$0.00 per request - PUT, COPY, POST, or LIST requests under the monthly global free tier	636.000 Requests	$0.00
Amazon Simple Storage Service Requests-Tier2		$0.00
$0.00 per request - GET and all other requests under the monthly global free tier	3,593.000 Requests	$0.00
Amazon Simple Storage Service TimedStorage-ByteHrs		$0.00
$0.000 per GB - storage under the monthly global free tier	0.009 GB-Mo	$0.00
Transcribe		**$0.00**
US East (N. Virginia)		**$0.00**
Amazon Transcribe StreamingAudio		$0.00
Processed streaming transcription request in us-east-1 - free tier	810.000 second	$0.00
Translate		**$0.00**
US East (N. Virginia)		**$0.00**
Amazon Translate TranslateText		$0.00
Processed translation request in US East (N. Virginia) - free-tier	1,398.000 Character	$0.00
Processed translation request in US East (N. Virginia) - free-tier	276.000 Character	$0.00
CloudWatch		**$0.00**
US East (N. Virginia)		**$0.00**
Amazon CloudWatch		$0.00
$0.00 per request - first 1,000,000 requests	203.000 Requests	$0.00
AmazonCloudWatch PutLogEvents		$0.00
First 5GB per month of log data ingested is free.	0.000131 GB	$0.00
Polly		**$0.00**
US East (N. Virginia)		**$0.00**
Amazon Polly USE1-SynthesizeSpeech-Characters		$0.00
First 5,000,000 characters (across regions) per month are free	1,896.000 Characters	$0.00
Amazon Polly USE1-SynthesizeSpeechNeural-Characters		$0.00
First 1,000,000 characters (across regions) per month are free for Neural TTS	148.000 Characters	$0.00

…even if you're already out of the FreeTier, this could cost as low as US $1

So it's almost...



THANK YOU!

CATHERINE CRUZ
DEVOPS ENGINEER
catherine.cruz@endava.com

CARLOS HERRERA
DEVOPS ENGINEER
carlos.herrera@endava.com