

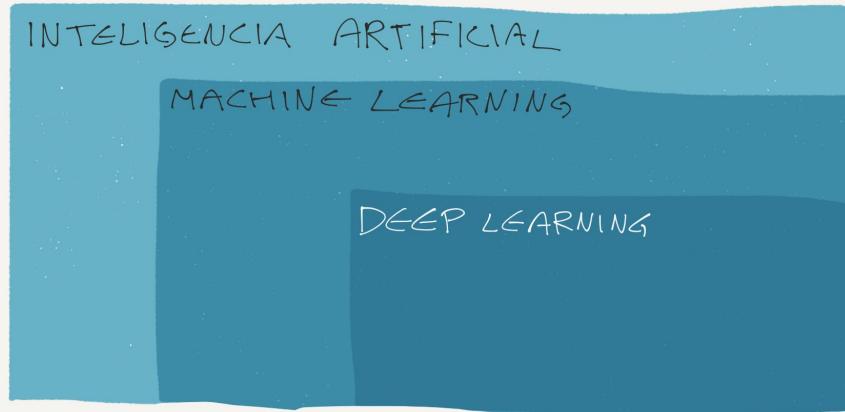


synapbox

OpenClass **AWS Rekognition**
por **Miguel Cabral**

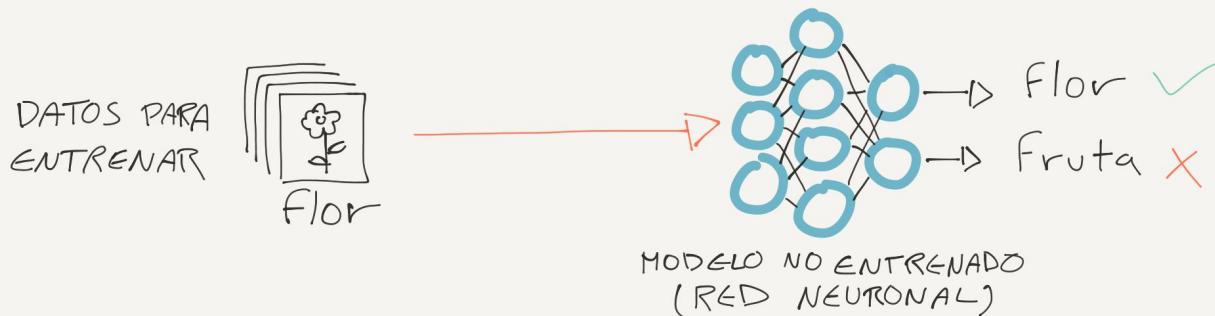
¿QUÉ ES DEEP LEARNING?

- SU OBJETIVO CONSISTE EN INFERRIR ABSTRACCIONES DE NIVEL SUPERIOR A PARTIR DE DATOS SIN PROCESAR USANDO UN GRÁFICO PROFUNDO CON NUMEROSES CAPAS DE PROCESAMIENTO COMPUSTAS POR MÚLTIPLES TRANSFORMACIONES LINEALES Y NO LINEALES (DEFINICIÓN DE AWS).

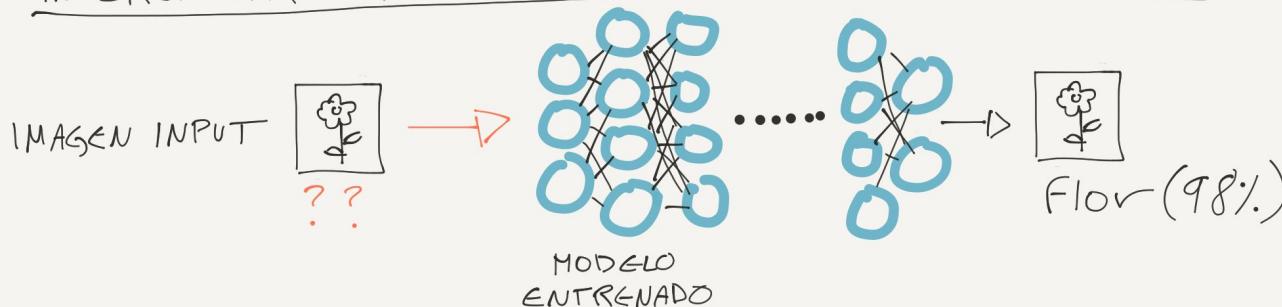


¿CÓMO FUNCIONA?

ENTRENAMIENTO (FASE APRENDIZAJE)



INFERENCIA (USO)



¿QUÉ NECESITAMOS?

- NODE.JS/NPM. [HTTPS://WWW.NPMJS.COM/GET-NPM](https://www.npmjs.com/get-npm)
- CUENTA "FREE TIER" DE AWS. [HTTPS://AWS.AMAZON.COM/FREE/](https://aws.amazon.com/free/)

AWS RECOGNITION

- FORMATOS JPG Y PNG
- TAMAÑOS 15MB COMO OBJETOS S3, 5MB COMO ARREGLO DE BYTES
- HOY USAREMOS NODE.JS PERO EXISTEN OTROS SDKS

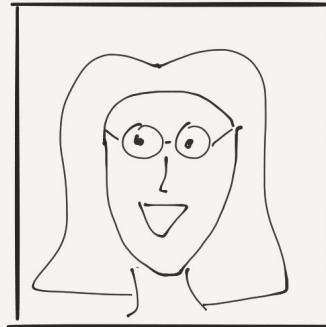
PROYECTO 1

ANUNCIOS SEGMENTADOS

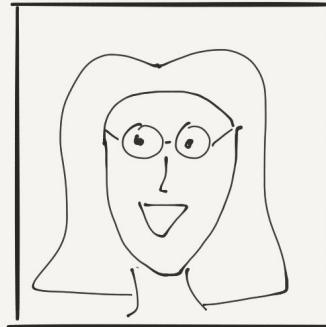
P1. ANUNCIOS SEGMENTADOS



DetectFaces API



DetectFaces API

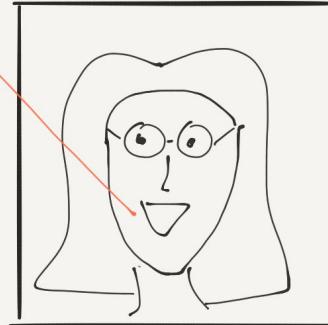


ANALISIS FACIAL

DetectFaces API

EMOCIONES

- Felicidad
- Sonriendo



ANALISIS FACIAL

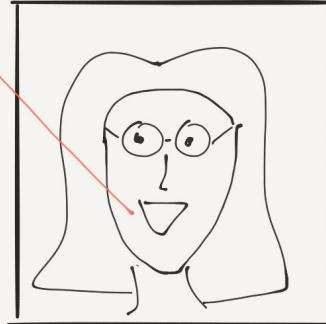
DetectFaces API

EMOCIONES

- Felicidad
- Sonriendo

DEMOGRAFÍA

- mujer
- edad: 20-30



ANALISIS FACIAL

DetectFaces API

EMOCIONES

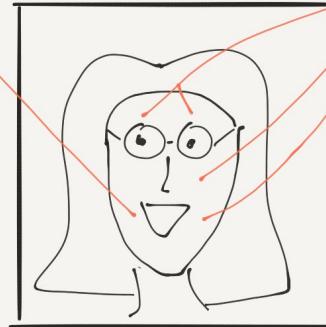
- Felicidad
- Sonriendo

DEMOGRAFÍA

- mujer
- edad: 20-30

REFERENCIAS FACIALES

- posición: ojos, nariz, boca, etc...



ANALISIS FACIAL

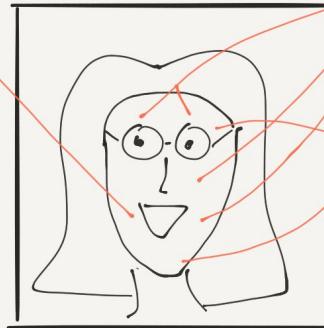
DetectFaces API

EMOCIONES

- Felicidad
- Sonriendo

DEMOGRAFÍA

- mujer
- edad: 20-30



REFERENCIAS FACIALES

- posición: ojos, nariz, boca, etc...

OTROS ATRIBUTOS

- lentes para ver y oscuros
- pelo facial (barba)
- posición y rotación de cara

ANALISIS FACIAL

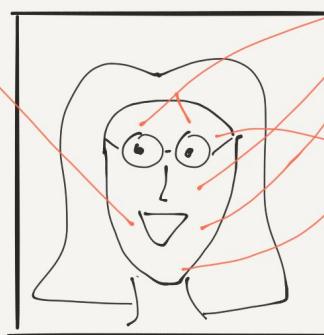
DetectFaces API

EMOCIONES

- Felicidad
- Sonriendo

DEMOGRAFÍA

- mujer
- edad: 20-30

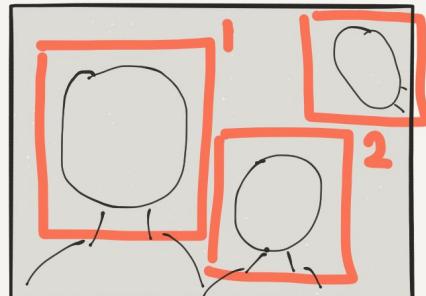


REFERENCIAS FACIALES

- posición: ojos, nariz, boca, etc...

OTROS ATRIBUTOS

- lentes para ver y oscuros
- pelo facial (barba)
- posición y rotación de cara



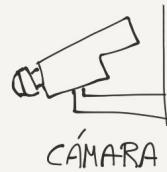
3
¡HASTA 100 CARAS!
ORDENA DE GRANDE A CHICA

ANALISIS FACIAL

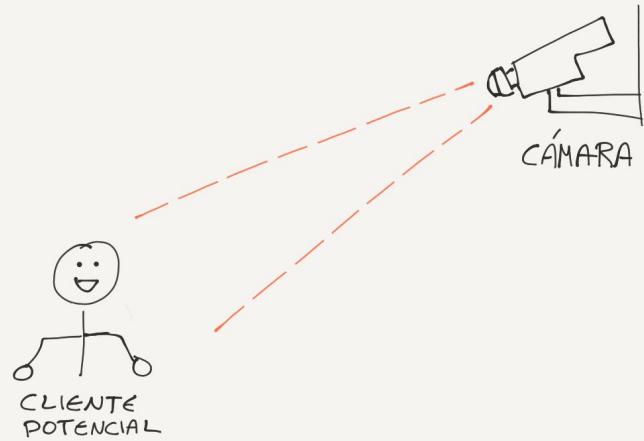
P1. ANUNCIOS SEGMENTADOS



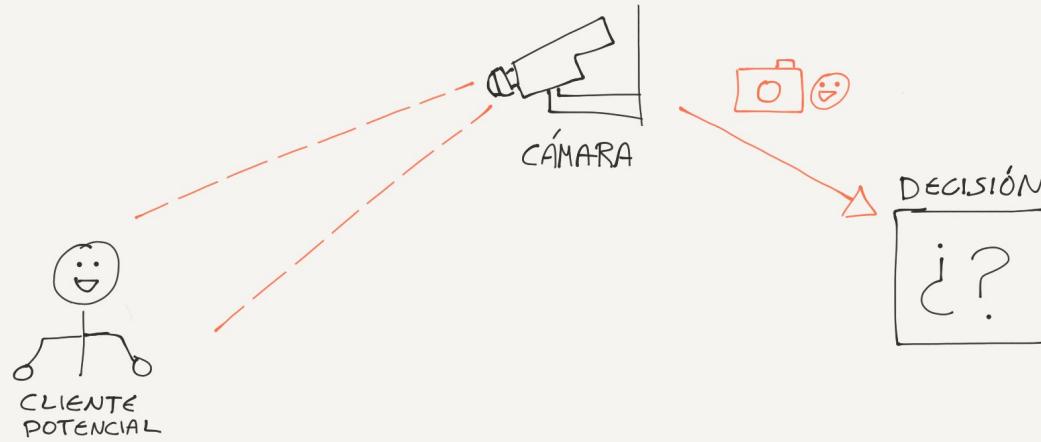
P1. ANUNCIOS SEGMENTADOS



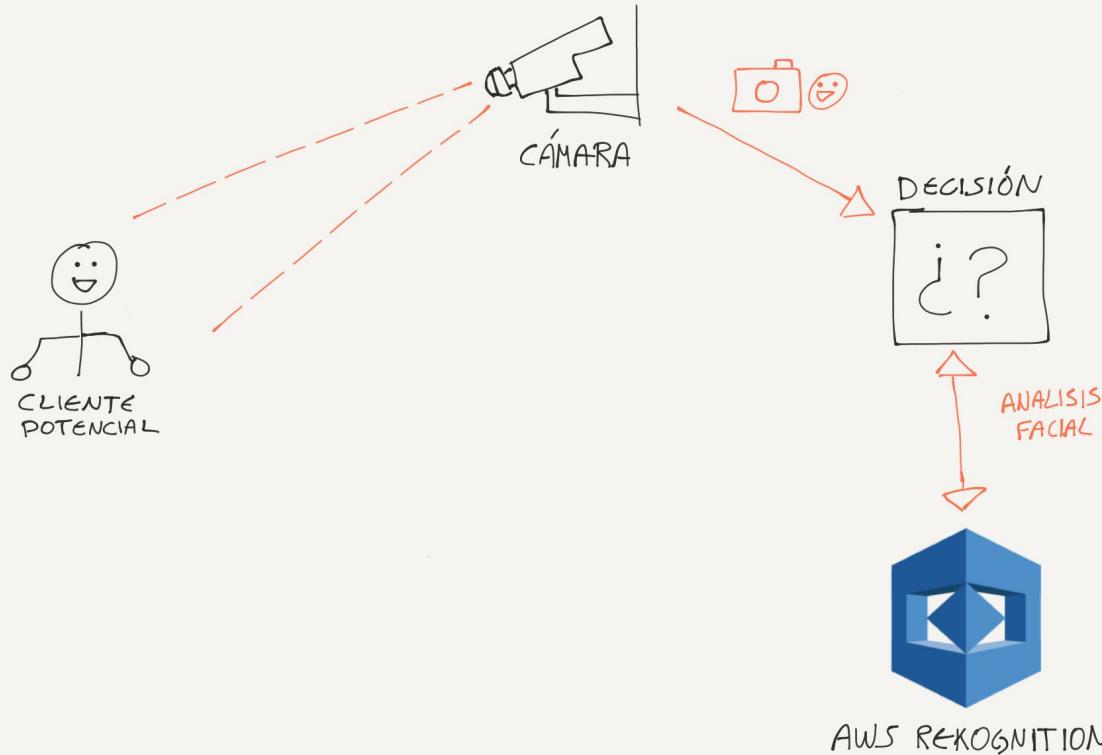
P1. ANUNCIOS SEGMENTADOS



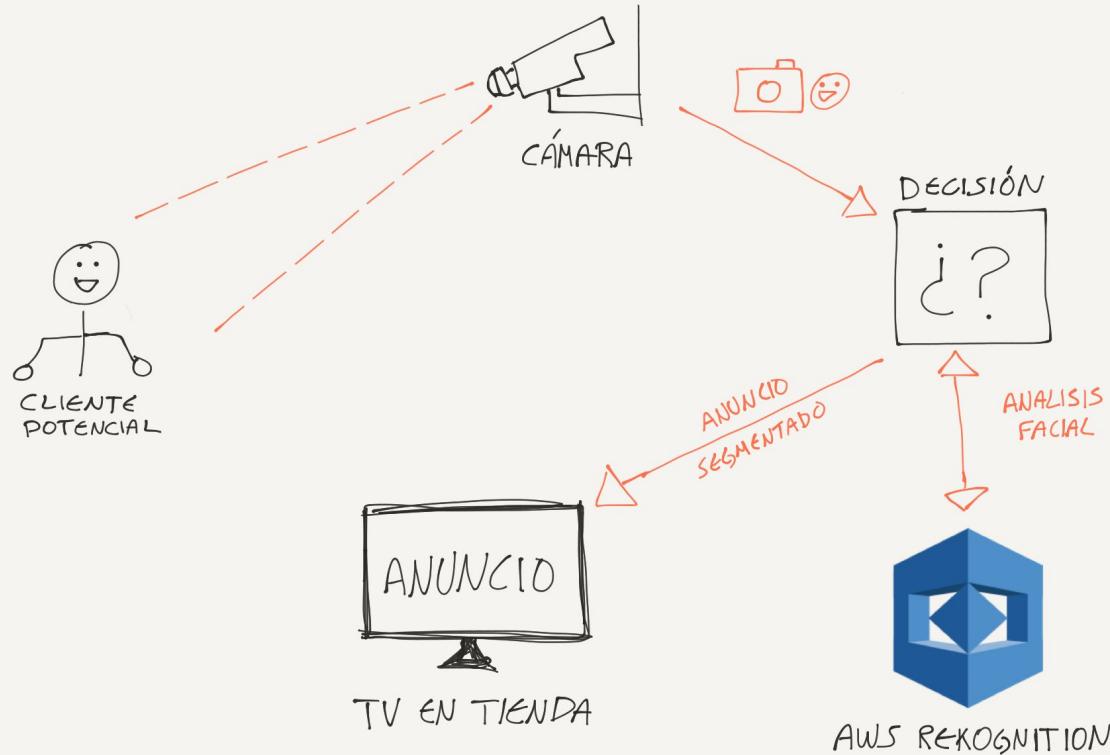
P1. ANUNCIOS SEGMENTADOS



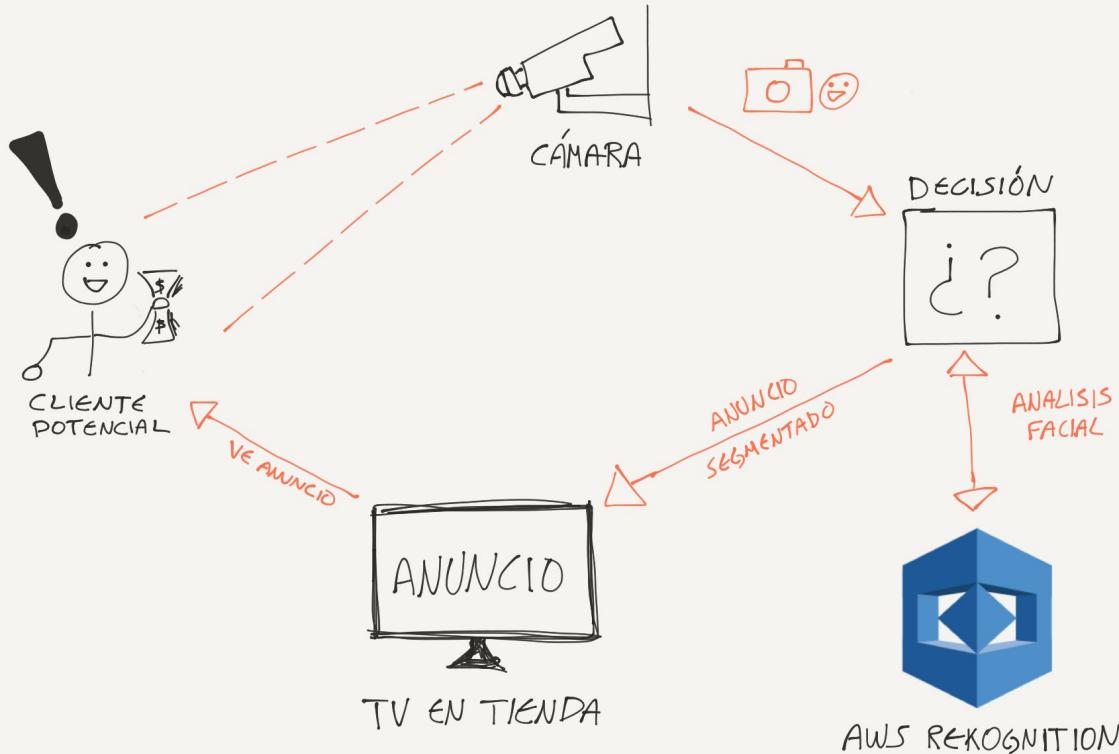
P1. ANUNCIOS SEGMENTADOS



P1. ANUNCIOS SEGMENTADOS



P1. ANUNCIOS SEGMENTADOS



SERVICIOS DE AWS A UTILIZAR



S3 BUCKET. SE UTILIZA PARA ALMACENAR CONTENIDO, COMO UNA CARPETA EN TU COMPUTADORA.



LAMBDA. FUNCIONES QUE SE DISPARAN CUANDO OCURRE ALGÚN EVENTO DE OTRO SERVICIO DE AWS.

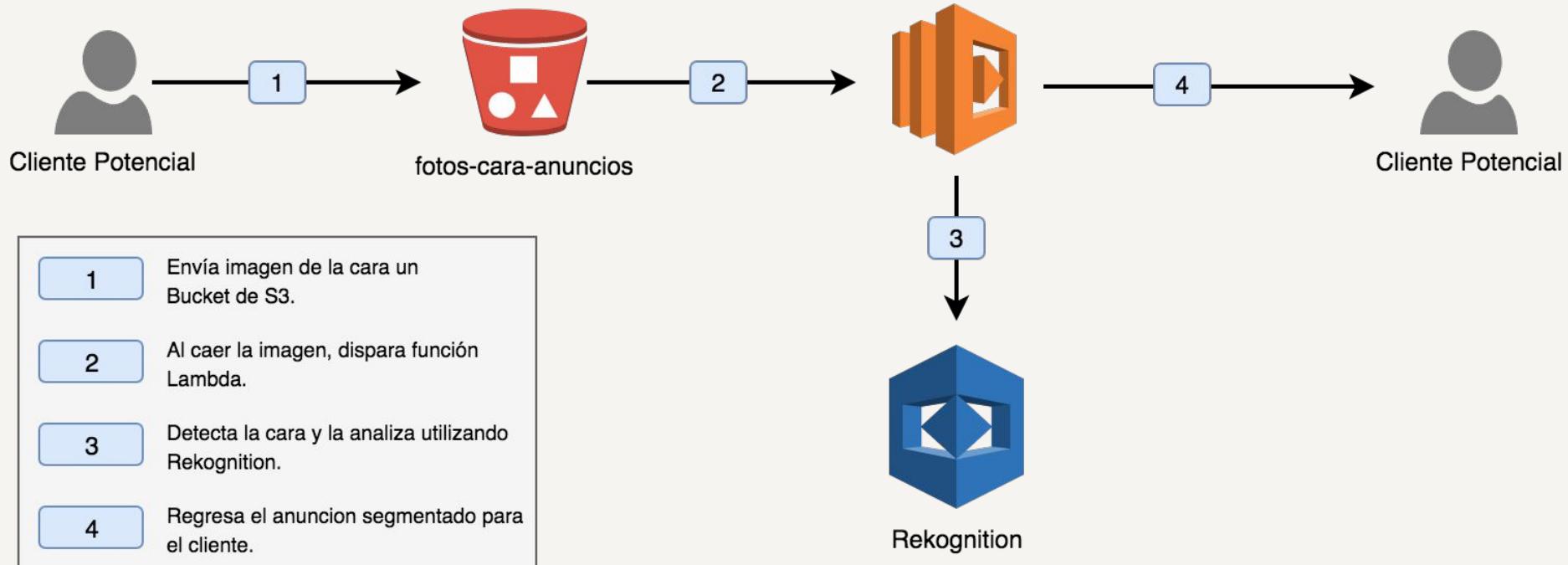


RECOGNITION!



API GATEWAY. USADO PARA CREAR Y MANEJAR APIs REST QUE EXPONEN LAMBDA Y OTROS SERVICIOS DE AWS.

ARQUITECTURA PARA ANUNCIOS SEGMENTADOS



CREAR BUCKETS S3

1. CREAR BUCKET ANUNCIOS-SEGMENTADOS* PARA ALMACENAR IMÁGENES DE ANUNCIOS.
 - A. AGREGAR BUCKET POLICY
 - B. SUBIR IMÁGENES DE ANUNCIOS
2. CREAR BUCKET DETECTA-CARA-PARA-ANUNCIOS* PARA ALMACENAR LA FOTO DE LOS CLIENTES.
 - A. AGREAR BUCKET POLICY
 - B. CONFIGURAR CORS POLICY

*EL NOMBRE DE LOS BUCKETS DEBE SER ÚNICO EN TODOS LADOS.

CREAR LAMBDA "BUSCARANUNCIO" PT.1

1. ABRIR EDITOR DE CÓDIGO Y CREAR CARPETA
2. CREAR PROYECTO EN NODE.JS USANDO LA TERMINAL
 - A. npm init
 - B. npm install aws-sdk --save
 - C. npm install http --save
2. CREAR index.js Y PEGAR CÓDIGO buscar-anuncio.js
3. CAMBIAR NOMBRE DE BUCKETS EN EL ARCHIVO

CREAR LAMBDA "BUSCARANUNCIO" PT. 2

1. CREAR ARCHIVO ZIP CON INDEX.JS, PACKAGE.JSON Y CARPETA NODE_MODULES
2. INGRESAR A CONSOLA DE AWS Y BUSCAR SERVICIO IAM
 - A. CREAR ROL PARA LAMBDA LLAMADO ROL-ANUNCIOS-SEGMENTADOS
 - B. ESCOGER POLICY AWSLAMBDAFULLACCESS Y AMAZONREKOGNITIONFULLACCESS

CREAR LAMBDA "BUSCARANUNCIO" PT. 3

1. BUSCAR SERVICIO AWS LAMBDA
2. CREAR LAMBDA BUSCARANUNCIO
 - A. SELECCIONAR ROL-ANUNCIOS-SEGMENTADOS
 - B. SUBIR ARCHIVO ZIP
 - C. INCREMENTAR TIMEOUT A 30 SEGUNDOS
 - D. GUARDAR
2. PROBAR LAMBDA
 - A. SUBIR IMAGEN DE PRUEBA A BUCKET DETECTA-CARA-PARA-ANUNCIOS
 - B. CONFIGURAR EVENTO DE PRUEBA CON FILENAME DE ARCHIVO SUBIDO
 - C. REVISAR LOGS EN CLOUDWATCH

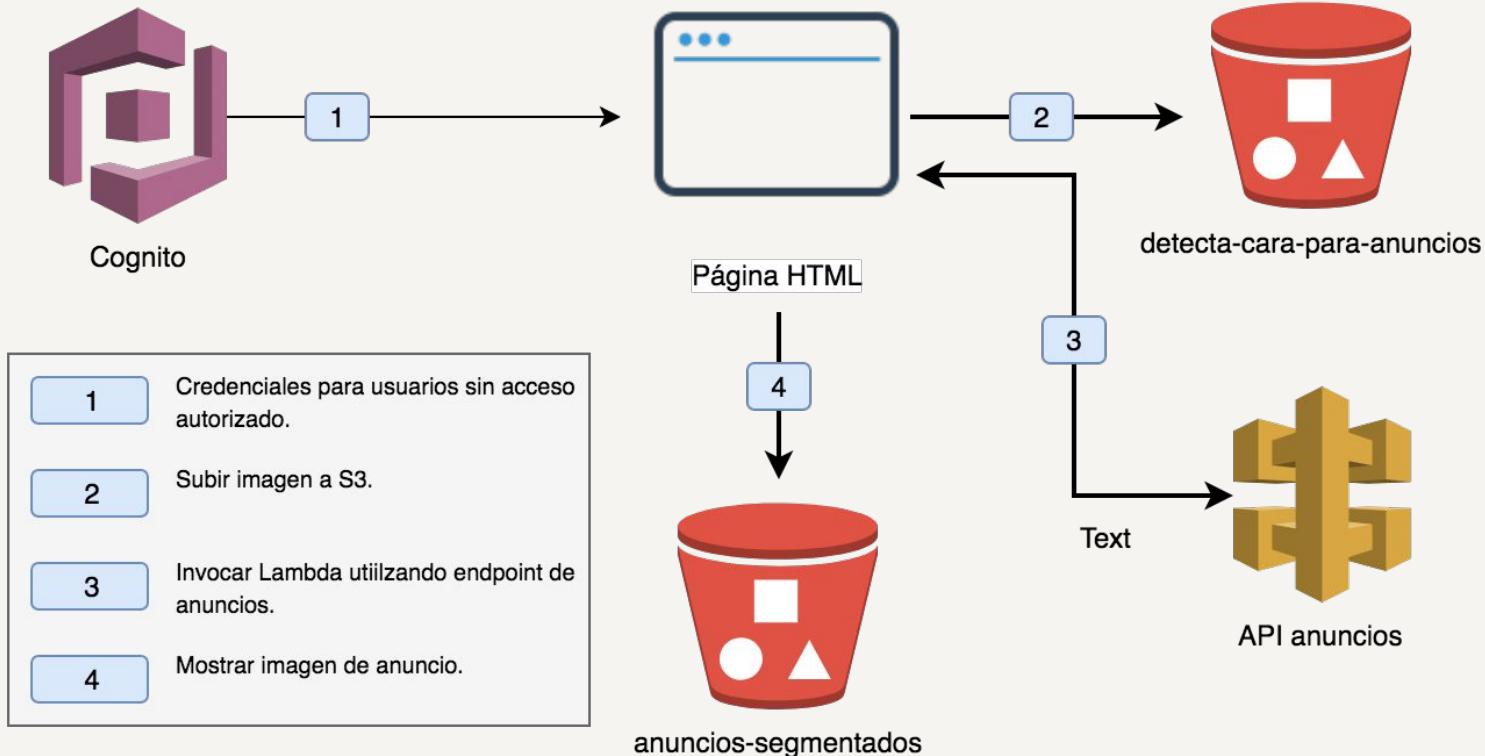
CREAR API PT.1

1. BUSCAR SERVICIO API GATEWAY Y CREAR API ANUNCIOS
2. CREAR MÉTODO GET
3. SELECCIONAR LAMBDA
4. ENTRAR A METHOD REQUEST
 - A. AGREGAR QUERY STRING FILENAME
5. ENTRAR A INTEGRATION REQUEST -> MAPPING TEMPLATES
 - A. SELECCIONAR WHEN THERE ARE NO TEMPLATES DEFINED (RECOMMENDED) Y AGREGAR TIPO DE CONTENIDO APPLICATION/JSON
 - B. AGREGAR TEMPLATE {"filename": "\$input.params('filename')"}

CREAR API PT. 2

1. SELECCIONAR ACTIONS Y ENABLE CORS
2. SELECCIONAR ACTIONS Y DEPLOY API
3. CREAR AMBIENTE
4. SELECCIONAR RESOURCES Y PROBAR USANDO FILENAME DE ARCHIVO SUBIDO PREVIAMENTE

FLUJO CLIENTE WEB



CREAR CLIENTE WEB PT.1

1. ABRIR ARCHIVO INDEX-ANUNCIOS.HTML
2. INGRESAR A CONSOLA DE AWS Y BUSCAR SERVICIO COGNITO
3. SELECCIONAR MANAGE IDENTITY POOLS
4. CREAR IDENTITY POOL Y DAR ACCESO A IDENTIDADES NO AUTENTICADAS
5. COPIAR IDENTITY POOL ID EN CLIENTE

CREAR CLIENTE WEB PT. 2

1. CAMBIAR NOMBRE DE BUCKET
2. INGRESAR A AWS Y BUSCAR SERVICIO API GATEWAY -> DASHBOARD PARA OBTENER URL DEL API
3. REEMPLAZAR EN CLIENTE
4. PROBAR CON IMÁGENES

CONCLUSIONES Y PREGUNTAS

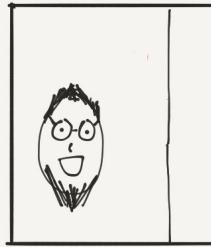
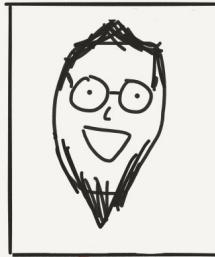
PROYECTO 1

PROYECTO 2

ACCESO Y AUTENTICACIÓN CON ROSTRO

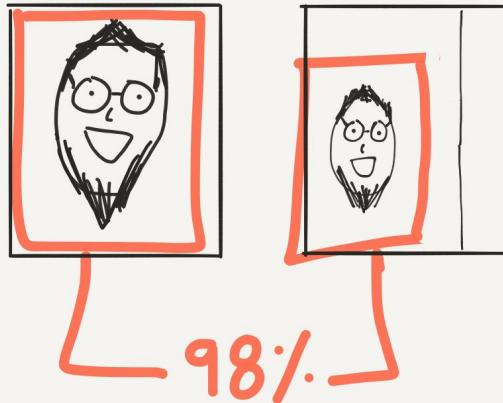
P2. ACCESO CON ROSTRO

CompareFacesAPI



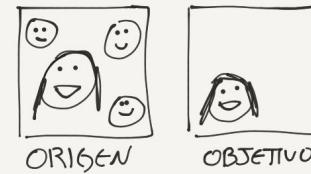
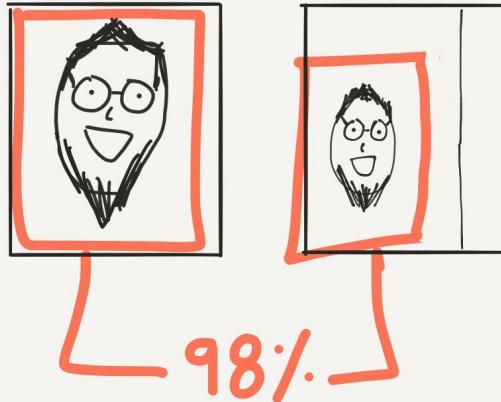
P2. ACCESO CON ROSTRO

CompareFacesAPI



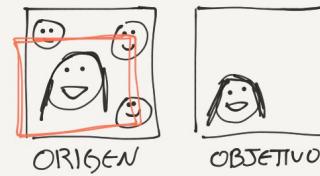
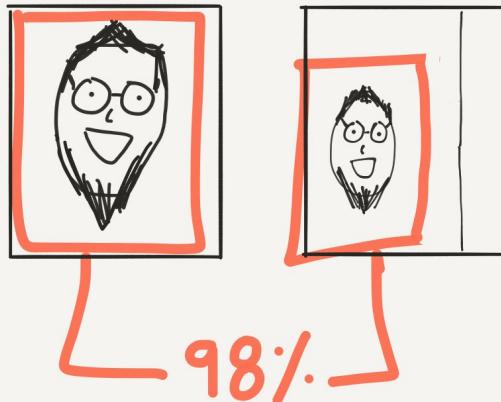
P2. ACCESO CON ROSTRO

CompareFacesAPI



P2. ACCESO CON ROSTRO

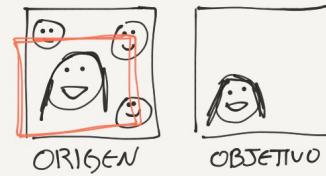
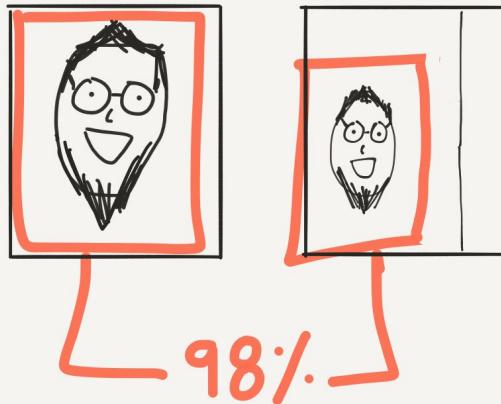
CompareFacesAPI



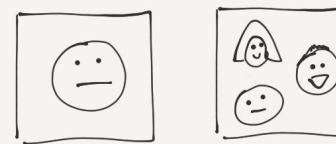
¡LA CARA
MÁS GRANDE!

P2. ACCESO CON ROSTRO

CompareFacesAPI

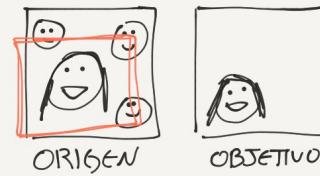
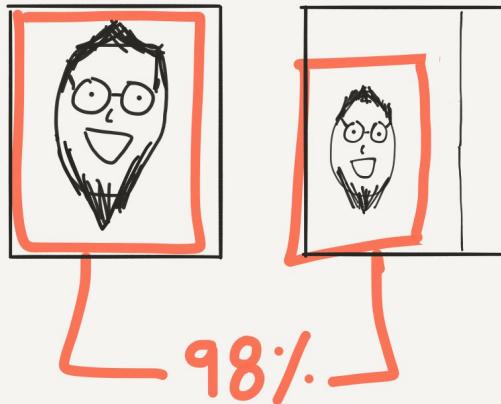


¡LA CARA
MÁS GRANDE!

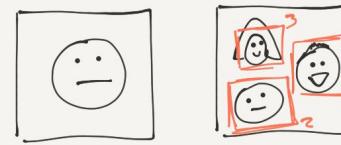


P2. ACCESO CON ROSTRO

CompareFacesAPI



¡LA CARA
MÁS GRANDE!

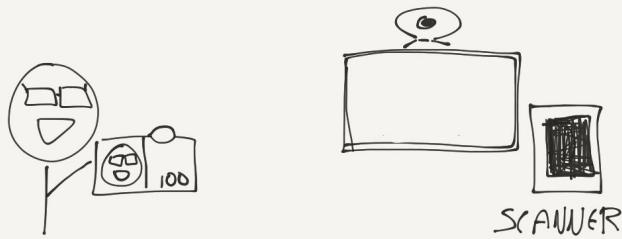


COMPARAR CONTRA
TODAS LAS CARAS
DE GRANDE A CHICA

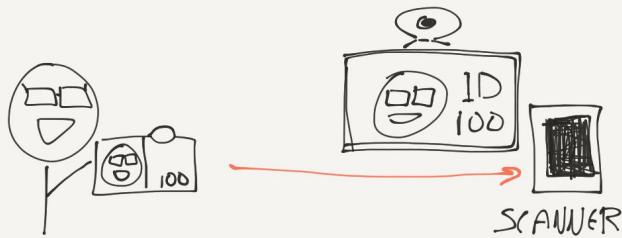
P2. ACCESO CON ROSTRO



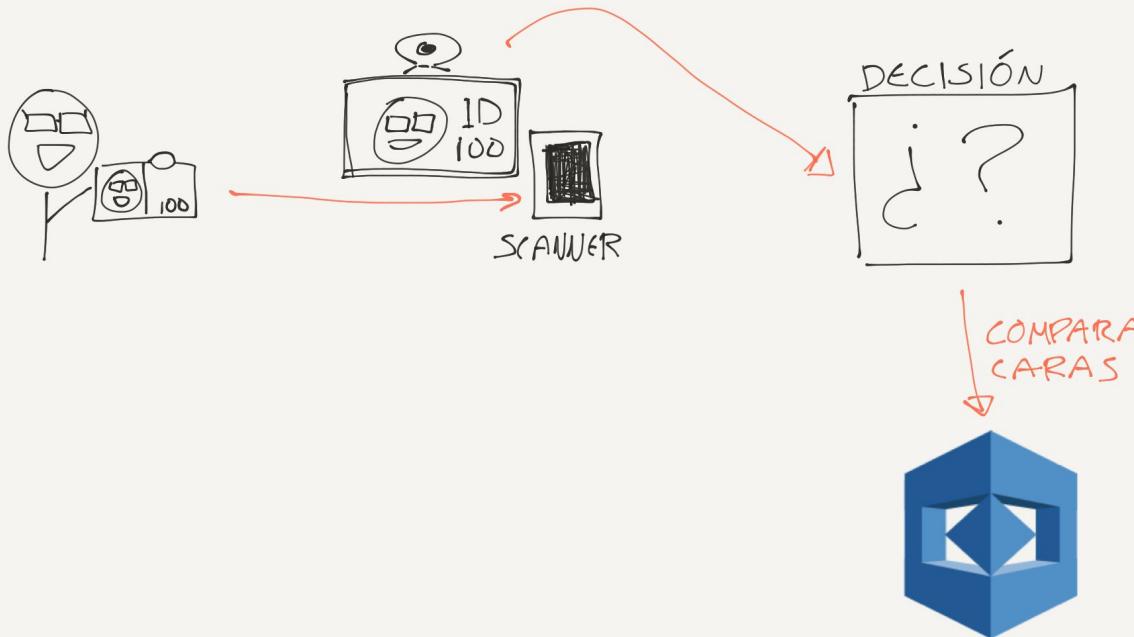
P2. ACCESO CON ROSTRO



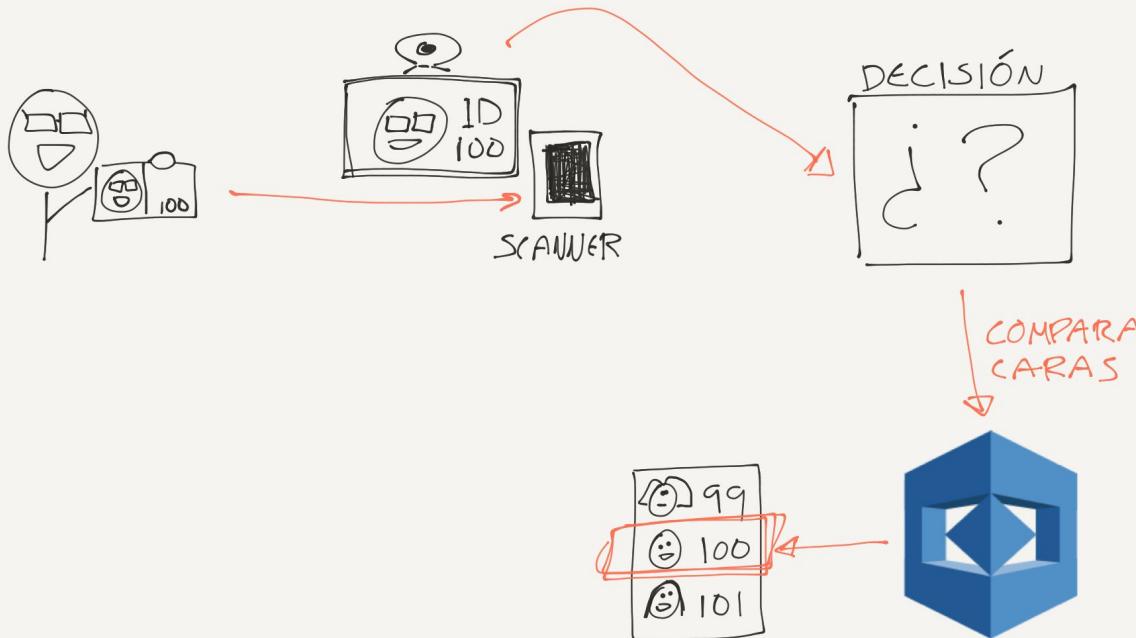
P2. ACCESO CON ROSTRO



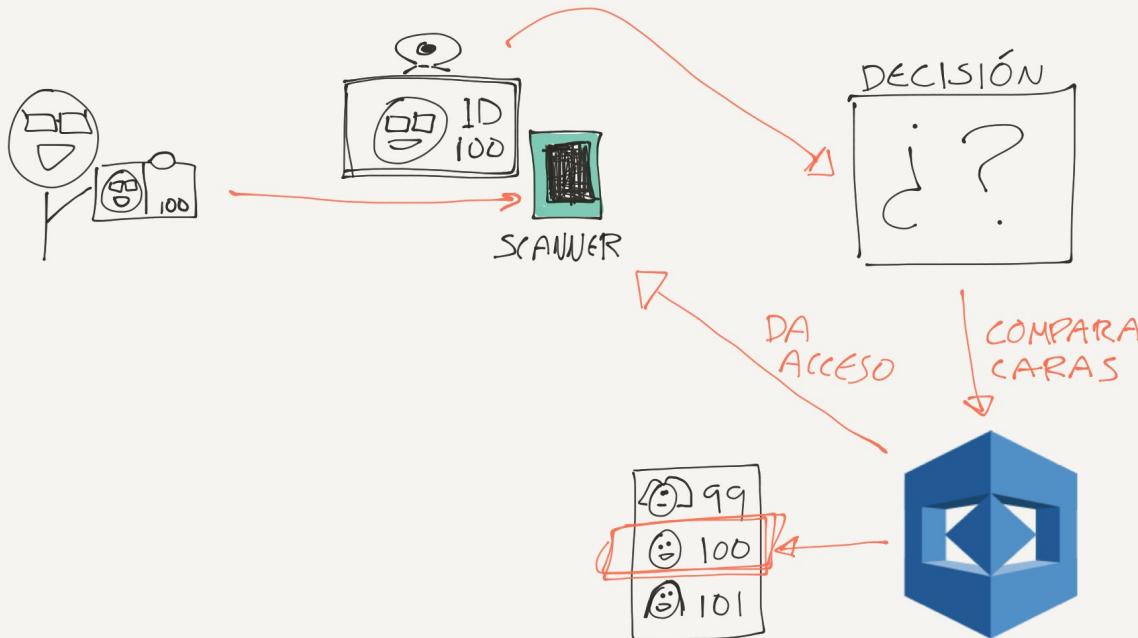
P2. ACCESO CON ROSTRO



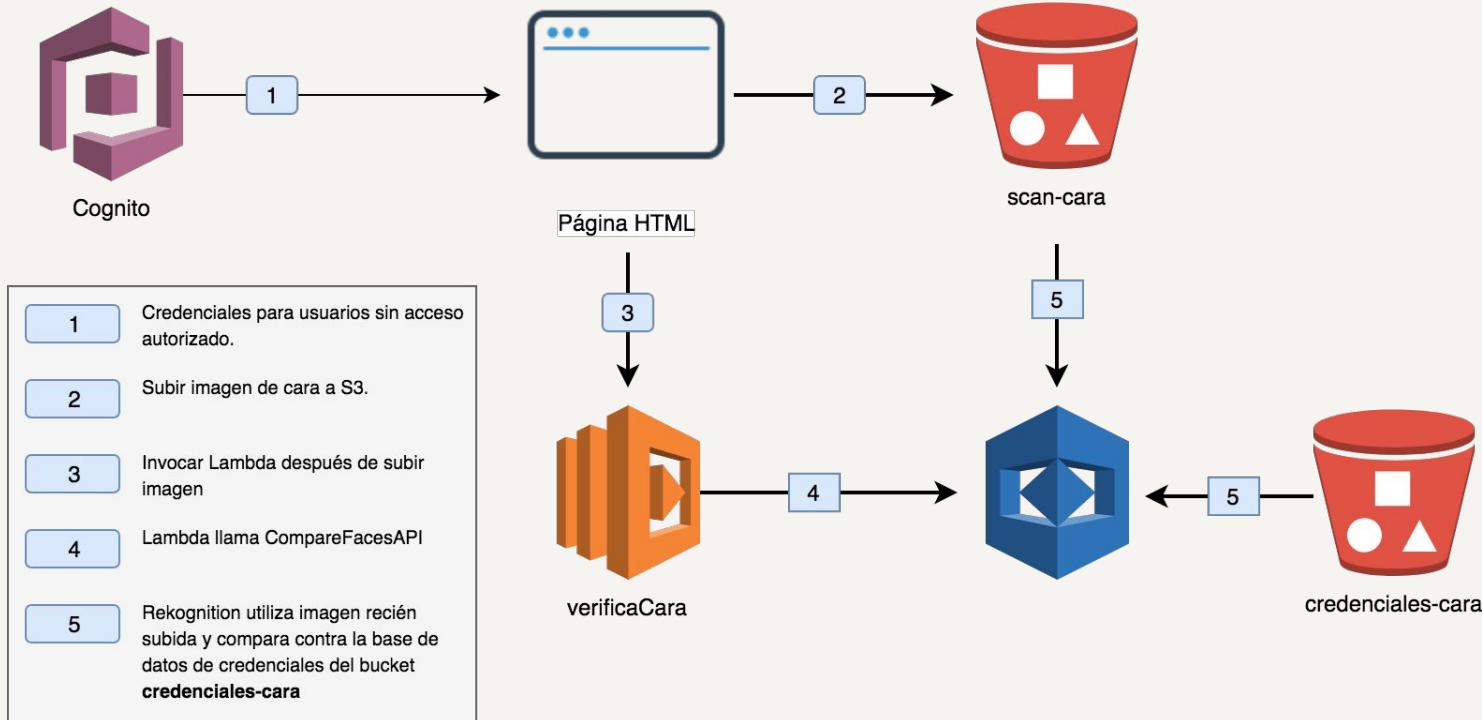
P2. ACCESO CON ROSTRO



P2. ACCESO CON ROSTRO



ARQUITECTURA PARA ACCESO CON ROSTRO



CREAR BUCKETS S3

1. CREAR BUCKET CREDENCIALES-CARA* PARA ALMACENAR IMÁGENES DE LAS CREDENCIALES.
 - A. AGREGAR BUCKET POLICY
 - B. SUBIR IMAGEN DE CREDENCIALES
2. CREAR BUCKET SCAN-CARA* PARA ALMACENAR LAS FOTOS QUE SE VAN A COMPARAR
 - A. AGREAR BUCKET POLICY
 - B. SUBIR IMAGEN COMPARE-1.JPG PARA PROBAR
 - C. CONFIGURAR CORS POLICY

*EL NOMBRE DE LOS BUCKETS DEBE SER ÚNICO EN TODOS LADOS.

CREAR LAMBDA "VERIFICACARA"

1. BUSCAR SERVICIO AWS LAMBDA
2. CREAR LAMBDA VERIFICACARA
 - A. SELECCIONAR ROL-ANUNCIOS-SEGMENTADOS
 - B. CAMBIAR NOMBRE BUCKETS
 - C. COPIAR CÓDIGO DE VERIFICA-CARA.JS Y PEGAR
 - D. INCREMENTAR TIMEOUT A 30 SEGUNDOS
2. PROBAR LAMBDA
 - A. CONFIGURAR EVENTO DE PRUEBA CON **filename** Y **idEmpleado**
 - B. REVISAR LOGS EN CLOUDWATCH

CREAR CLIENTE WEB PT.1

1. ABRIR ARCHIVO INDEX-VERIFYCAR.HTML
2. INGRESAR A CONSOLA DE AWS Y BUSCAR SERVICIO COGNITO
3. SELECCIONAR MANAGE IDENTITY POOLS
4. CREAR IDENTITY POOL Y DAR ACCESO A IDENTIDADES NO AUTENTICADAS
5. SELECCIONAR VIEW POLICY DOCUMENT EN ROLE SUMMARY
6. EDITAR DOCUMENTO Y AGREGAR **lambda : InvokeFunction**
7. COPIAR IDENTITY POOL ID EN CLIENTE

CREAR CLIENTE WEB PT.2

1. CAMBIAR NOMBRE DE BUCKETS
2. CAMBIAR NOMBRE DE FUNCIÓN LAMBDA
3. PROBAR CON IMÁGENES Y IDS DE EMPLEADO

CONCLUSIONES Y PREGUNTAS

PROYECTO 2