



“SISTEMA PARA EL MONITOREO, DETECCIÓN Y ALERTA DE  
SOMNOLENCIA DEL CONDUCTOR MEDIANTE VISIÓN ARTIFICIAL,  
COMUNICACIÓN INALÁMBRICA Y GEOLOCALIZACIÓN”

---

## Segundo Reporte Parcial

---

### Lista de actividades

- Maquetación web
- Investigación de la documentación del módulo 3G/4G LTE-Base Hat
- Enlace de Amazon S3 con el sistema backend
- Creación de los servicios backend
- Familiarizarse con el entorno de desarrollo de la NVIDIA Jetson Nano
- Implementar algoritmo para la detección del rostro y ojos
- Implementar puntos faciales en el rostro y métrica MOR

### *Autores:*

Alan Eduardo Gamboa Del  
Ángel  
Maite Paulette Díaz Martínez

### *Asesores:*

M.en C. Niels Henrik Navarrete  
Manzanilla  
Dr. Rodolfo Vera Amaro

13 de Marzo 2023

# Índice

<b>1. Maquetación web</b>	<b>4</b>
1.1. Objetivo . . . . .	4
1.2. Descripción . . . . .	4
1.3. Resultados . . . . .	7
<b>2. Investigación de la documentación del módulo 3G/4G LTE-Base Hat</b>	<b>8</b>
2.1. Objetivo . . . . .	8
2.2. Descripción . . . . .	8
2.3. Resultados . . . . .	10
<b>3. Enlace de Amazon S3 con el sistema backend</b>	<b>12</b>
3.1. Objetivo . . . . .	12
3.2. Descripción . . . . .	12
3.3. Resultados . . . . .	14
<b>4. Creación de los servicios backend</b>	<b>15</b>
4.1. Objetivo . . . . .	15
4.2. Descripción . . . . .	15
4.3. Resultados . . . . .	15
<b>5. Familiarizarse con el entorno de desarrollo de la NVIDIA Jetson Nano</b>	<b>17</b>
5.1. Objetivo . . . . .	17
5.2. Descripción . . . . .	17
5.3. Resultados . . . . .	18
<b>6. Implementar algoritmo para la detección del rostro y ojos</b>	<b>19</b>
6.1. Objetivo . . . . .	19
6.2. Descripción . . . . .	19
6.3. Resultados . . . . .	19
<b>7. Implementar puntos faciales en el rostro y la metrica MOR</b>	<b>20</b>
7.1. Objetivo . . . . .	20
7.2. Descripción . . . . .	20
7.3. Resultados . . . . .	20
<b>8. Conclusiones</b>	<b>21</b>
<b>9. Bibliografía</b>	<b>22</b>

## Índice de figuras

1.	Creación proyecto de react . . . . .	4
2.	Instalación React Router DOM . . . . .	4
3.	Función RequireAuth . . . . .	5
4.	Componente Login . . . . .	5
5.	Ruta protegida del componente Home . . . . .	6
6.	Directorio de rutas . . . . .	6
7.	Página de Login . . . . .	7
8.	Instalación Amplify CLI . . . . .	8
9.	Instalación de librerías . . . . .	8
10.	Implementación del SDK de Amplify . . . . .	9
11.	Creación de aplicación de Express . . . . .	9
12.	Amplify add api . . . . .	9
13.	Configuración de parámetros de GraphQL . . . . .	9
14.	Definición de Schemas para el manejo de datos . . . . .	10
15.	Generación de Endpoint de GraphQL . . . . .	11
16.	Configuración de servicio de almacenamiento S3 . . . . .	12
17.	Generación de Endpoint de GraphQL . . . . .	12
18.	Generación de Endpoint de GraphQL . . . . .	12
19.	Generación de Endpoint de GraphQL . . . . .	13
20.	Generación de Endpoint de GraphQL . . . . .	13
21.	Generación de Endpoint de GraphQL . . . . .	13
22.	Generación de Endpoint de GraphQL . . . . .	14
23.	Tablas generadas mediante los schemas definidos . . . . .	14
24.	Funcionamiento de Appsync . . . . .	15
25.	Generación de endpoint de GraphQL . . . . .	16
26.	Funcionamiento de Amazon Cognito . . . . .	18

## Índice de tablas

# 1. Maquetación web

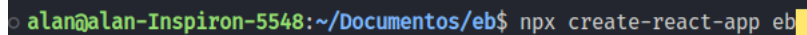
## 1.1. Objetivo

Definir e implementar las rutas que tendrá la aplicación, así como si serán públicas o privadas y la información que se desplegará en cada una de las mismas.

## 1.2. Descripción

### Rutas públicas vs rutas protegidas

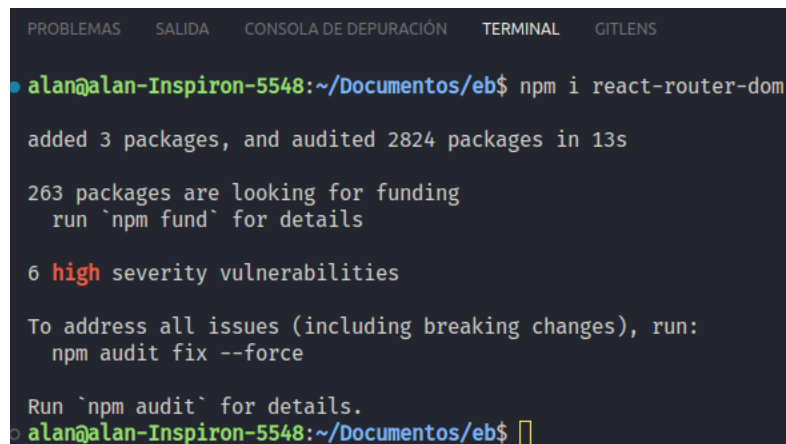
Cuando se habla de una ruta protegida en React, se refiere a programar un bloqueo en ciertas rutas a la cual se le restringe el acceso al usuario. Esto comunmente se realiza para la validación de inicio de sesión de usuarios. Sí el usuario no tiene una sesión iniciada, no podrá acceder a las rutas protegidas de la aplicación. Por otro lado, las rutas públicas son todas aquellas las cuales no requieren contar con una sesión iniciada, y pueden ser accesadas por cualquier tipo de usuario[1]. Como primer paso, se necesita crear un proyecto de React utilizando el siguiente comando:



```
alan@alan-Inspiron-5548:~/Documentos/eb$ npx create-react-app eb
```

Figura 1: Creación proyecto de react

Posteriormente, se realiza la instalación del moudelo *React Router Dom* utilizando el siguiente comando:



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  GITLENS

alan@alan-Inspiron-5548:~/Documentos/eb$ npm i react-router-dom

added 3 packages, and audited 2824 packages in 13s

263 packages are looking for funding
  run `npm fund` for details

6 high severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
alan@alan-Inspiron-5548:~/Documentos/eb$
```

Figura 2: Instalación React Router DOM

Utilizando la librería *useAuthenticator* ofrecida por el paquete de React Dom, crearemos un archivo de nombre *RequireAuth.js* el cuál contendrá una función la cual se encargará de validar si existe una sesión iniciada previamente.

```

src > RequireAuth.js > RequireAuth
1  import { useLocation, Navigate } from 'react-router-dom';
2  import { useAuthenticator } from '@aws-amplify/ui-react';
3
4  export function RequireAuth({ children }) {
5    const location = useLocation();
6    const { route } = useAuthenticator((context) => [context.route]);
7    if (route !== 'authenticated') {
8      return <Navigate to="/login" state={{ from: location }} replace />;
9    }
10   return children;
11 }

```

Figura 3: Función RequireAuth

Posteriormente, se necesita contar con una página de Login, la cuál permitirá validar que se encuentre una sesión iniciada por parte del usuario, para así poder acceder a las rutas protegidas.

```

src > Login.js > ...
1  import { useEffect } from "react";
2  import { Authenticator, useAuthenticator, View } from '@aws-amplify/ui-react';
3  import '@aws-amplify/ui-react/styles.css';
4  import { useNavigate, useLocation } from 'react-router';
5
6  export function Login() {
7    const { route } = useAuthenticator((context) => [context.route]);
8    const location = useLocation();
9    const navigate = useNavigate();
10   let from = location.state?.from?.pathname // '/';
11   useEffect(() => {
12     if (route === 'authenticated') {
13       navigate(from, { replace: true });
14     }
15   }, [route, navigate, from]);
16   return (
17     <View className="auth-wrapper">
18       <Authenticator></Authenticator>
19     </View>
20   );
21 }

```

Figura 4: Componente Login

Para indicar a React, que se desea implementar una ruta protegida, se necesita ir al componente de dicha ruta e ingresar el siguiente código:

```

src > Home.js > Home
1  import { useAuthenticator, Authenticator } from "@aws-amplify/ui-react";
2
3  export default function Home() {
4    const { route } = useAuthenticator((context) => [context.route]);
5    const message =
6      route === 'authenticated' ? 'FIRST PROTECTED ROUTE!' : 'Loading ...';
7    return (
8      <Authenticator>
9        ({ { signOut, user } }) => (
10          <main>
11            <h1>Bienvenido a Home</h1>
12
13            <button onClick={signOut}>Sign out</button>
14          </main>
15        )
16      </Authenticator>
17    );
18
19  }
20

```

Figura 5: Ruta protegida del componente Home

En dicho componente, se hace uso de las librerías *useAuthenticator* y *Authenticator* las cuales son ofrecidas por los servicios de Amazon Amplify. Se tendrá que hacer esto para todos los componentes que deseemos mantener como rutas protegidas.

Finalmente, dentro de nuestro componente **App.js**, crearemos una función que contendrá el directorio de rutas tanto públicas como protegidas:

```

src > App.js > MyRoutes
37
38  export function MyRoutes(){
39    return (
40      <BrowserRouter>
41        <Routes>
42          <Route path="/" element={<Layout />} />
43          <Route index element={<Home />} />
44          <Route
45            path="/conductor"
46            element={
47              <RequireAuth>
48                <Conductor />
49              </RequireAuth>
50            }
51          />
52          <Route
53            path="/incidencia"
54            element={
55              <RequireAuth>
56                <Incidencia />
57              </RequireAuth>
58            }
59          />
60
61          <Route
62            path="/ubicacion"
63            element={

```

Figura 6: Directorio de rutas

Las rutas protegidas, estarán dentro de las etiquetas `<RequireAuth>`/`</RequireAuth>`, mientras que las públicas, irán dentro de las etiquetas `<Route>`/`</Route>`.

### 1.3. Resultados

Como resultado de todo lo anterior, tendremos una página de Login que utilizando los servicios de AWS Amplify y Cognito, permitirá iniciar sesión así como registrar a nuevos usuarios.

Home Conductor Incidencia Login

login

or favor inicia sesión

Registrarse Regístrate

Email

Ingresa tu Email

Contraseña

Ingresa tu Contraseña

Sign in

Forgot your password?

Figura 7: Página de Login

Si intentamos ingresar a las paginas de Conductores, Incidencias o Ubicacion, nos redigirá a la página de Login, debido a que estas páginas fueron definidas como rutas protegidas. Por lo tanto el usuario debe haber iniciado sesión para poder acceder a las mismas.

- **Path:** /  
**Descripción:** En esta dirección se encontrará la el formulario para poder iniciar sesión o registrarse
- **Path:** /home  
**Descripción:** Esta dirección será la página principal de la aplicación dónde se mostrarán las incidencias más recientes así como una lista de todos los conductores
- **Path:** /conductor/  
**Descripción:** Esta dirección mostrará el perfil del conductor de id correspondiente
- **Path:** /detalle\_incidencia  
**Descripción:** Esta dirección mostrará cada incidencia mostrando detalles como hora, fecha, coordenadas
- **Path:** /conductor/id/ubicacion  
**Descripción:** En esta vista se mostrará la ubicación en tiempo real de cada conductor
- **Path:** /conductor/id/incidencias  
**Descripción:** En esta vista se mostrará todas las incidencias registradas por cada conductor



## 2. Investigación de la documentación del módulo 3G/4G LTE-Base Hat

### 2.1. Objetivo

Familiarizarse con las distintas funciones y comandos, así como el entorno de desarrollo que ofrece el dispositivo Base-Hat

### 2.2. Descripción

Para el presente proyecto, se hará uso del Base-Hat SIM7600G-H 4G para Jetson Nano.



Figura 8: Instalación Amplify CLI

En primer lugar, utilizando la terminal del sistema operativo Ubuntu, se ingresan los siguientes comandos:

```
sudo apt-get update
sudo apt-get install python3-pip
sudo pip3 install pyserial
mkdir -p ~/Documents/SIM7600X_4G_for_JETSON_NANO
wget -P ~/Documents/SIM7600X_4G_for_JETSON_NANO/ https://www.waveshare.com/w/upload/6/64/SIM7600X_4G_for_JETSON_NANO.tar.gz
cd ~/Documents/SIM7600X_4G_for_JETSON_NANO/
tar -xvf SIM7600X_4G_for_JETSON_NANO.tar.gz
sudo pip3 install Jetson.GPIO
sudo groupadd -f -r gpio
sudo usermod -a -G gpio your_user_name
sudo udevadm control --reload-rules && sudo udevadm trigger
sudo apt-get install minicom
```

Figura 9: Instalación de librerías

Los comandos ingresados en la figura 9

Utilizando un editor de código, se necesita especificar que estará utilizando el *SDK* de Amazon Amplify:

```
const AWS = require('aws-sdk')
const awsServerlessExpressMiddleware = require('aws-serverless-express/middleware')
const bodyParser = require('body-parser')
const express = require('express')

AWS.config.update({ region: process.env.TABLE_REGION });
```

Figura 10: Implementación del SDK de Amplify

Posteriormente, declaramos una aplicación de ExpressJs la cuál nos permitirá ejecutar entre otras cosas, peticiones HTTP para la comunicación con la base de datos.

```
// declaracion de una app de express
const app = express()
app.use(bodyParser.json())
app.use(awsServerlessExpressMiddleware.eventContext())
```

Figura 11: Creación de aplicación de Express

Finalmente, se necesitan definir las rutas de las APIs que se estarán utilizando en el proyecto, las cuales serán dos, la primera se encargará de la aplicación web, y la segunda de realizar la comunicación con el Módulo Central de Procesamiento.

Para agregar una api, se necesita ejecutar el siguiente comando desde el directorio raíz del proyecto.

```
alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add api
```

Figura 12: Amplify add api

La consola de AWS Amplify requerirá introducir parámetros con los cuales serán construida nuestra API, para el presente proyecto se estará utilizando una arquitectura mediante GraphQL, por lo cual seleccionaremos dicha opción.

```
alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add api
? Select from one of the below mentioned services: GraphQL
? Here is the GraphQL API that we will create. Select a setting to edit or continue Continue
? Choose a schema template: One-to-many relationship (e.g., "Blogs" with "Posts" and "Comments")

⚠ WARNING: your GraphQL API currently allows public create, read, update, and delete access to all models via a n API Key. To configure PRODUCTION-READY authorization rules, review: https://docs.amplify.aws/cli/graphql/authorization-rules

✔ GraphQL schema compiled successfully.

Edit your schema at /home/alan/Documentos/eb/amplify/backend/api/ebase/schema.graphql or place .graphql files in a directory at /home/alan/Documentos/eb/amplify/backend/api/ebase/schema
✓ Do you want to edit the schema now? (Y/n) · yes
```

Figura 13: Configuración de parámetros de GraphQL

Antes de poder publicar nuestra API en AWS Amplify, se necesita definir el Schema con el cuál se hará el manejo de datos. A continuación se muestran dichos schemas:

```
amplify > backend > api > ebase > ✎ schema.graphql
1  # This "input" configures a global authorization rule to enable
2  # all models in this schema. Learn more about authorization rule
3  input AMPLIFY { globalAuthRule: AuthRule = { allow: public } } #
4
5  type Conductor @model {
6    id: ID
7    nombre: String
8    apellido: String
9    incidencias: [Incidencia] @hasMany
10   num_incidencias: Int
11  }
12
13  type Incidencia @model {
14    id: ID
15    title: String
16    estado: Boolean
17    conductor: Conductor @belongsTo
18    detalles: [Detalles] @hasOne
19    fecha_hora: Date
20  }
21
22  type Detalles @model {
23    id: ID
24    incidencia: Incidencia @belongsTo
25    ubicacion: String
26    url_video: String
27  }
28
29
```

Figura 14: Definición de Schemas para el manejo de datos

Finalmente, ejecutaremos el comando *amplify push*, el cuál publicará la API hacia AWS amplify, generando el endpoint correspondiente a dicha API

## 2.3. Resultados

Como resultado, tenemos el endpoint de nuestro modelo de GraphQL y nuestra API Key generada por Amplify

```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  GITLENS

Deployment completed.
Deploying root stack ebase [ ===== ] 1/3
  amplify-ebase-dev-175126    AWS::CloudFormation::Stack  UPDATE_IN_PROGRESS
  apiebase                    AWS::CloudFormation::Stack  CREATE_IN_PROGRESS
  authebase35f92a6b          AWS::CloudFormation::Stack  UPDATE_COMPLETE
Deployed api ebase [ ===== ] 9/9
  GraphQLAPI                  AWS::AppSync::GraphQLApi    CREATE_COMPLETE
  GraphQLAPINONEDS95A13CF0    AWS::AppSync::DataSource    CREATE_COMPLETE
  GraphQLAPIDefaultApiKey215A6D... AWS::AppSync::ApiKey        CREATE_COMPLETE
  GraphQLAPITransformerSchema3C... AWS::AppSync::GraphQLSchema CREATE_COMPLETE
  Blog                        AWS::CloudFormation::Stack  CREATE_COMPLETE
  Comment                     AWS::CloudFormation::Stack  CREATE_COMPLETE
  Post                        AWS::CloudFormation::Stack  CREATE_COMPLETE
  ConnectionStack             AWS::CloudFormation::Stack  CREATE_COMPLETE
  CustomResourcesjson         AWS::CloudFormation::Stack  CREATE_COMPLETE

✓ Generated GraphQL operations successfully and saved at src/graphql
Deployment state saved successfully.

GraphQL endpoint: [REDACTED].appsync-api.us-east-1.amazonaws.com/graphql
GraphQL API KEY: ([REDACTED])jm

GraphQL transformer version: 2

alan@alan-Inspiron-5548:~/Documentos/eb$
```

Figura 15: Generación de Endpoint de GraphQL

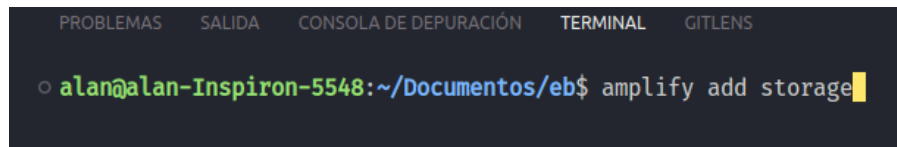
### 3. Enlace de Amazon S3 con el sistema backend

#### 3.1. Objetivo

Configurar e implementar la comunicación entre el sistema de alojamiento Amazon S3 y el sistema backend

#### 3.2. Descripción

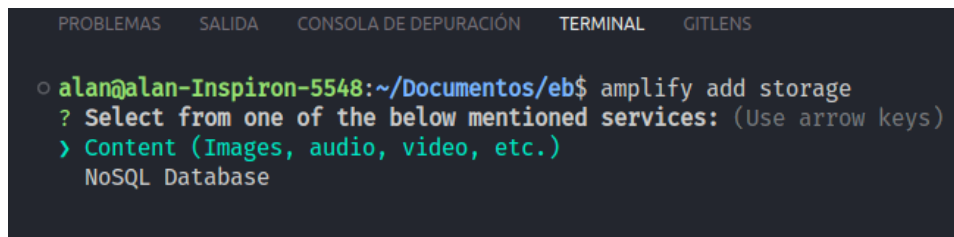
Utilizando un editor de código, y desde el directorio raíz de la aplicación, se deberá introducir el siguiente comando:



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  GITLENS
alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
```

Figura 16: Configuración de servicio de almacenamiento S3

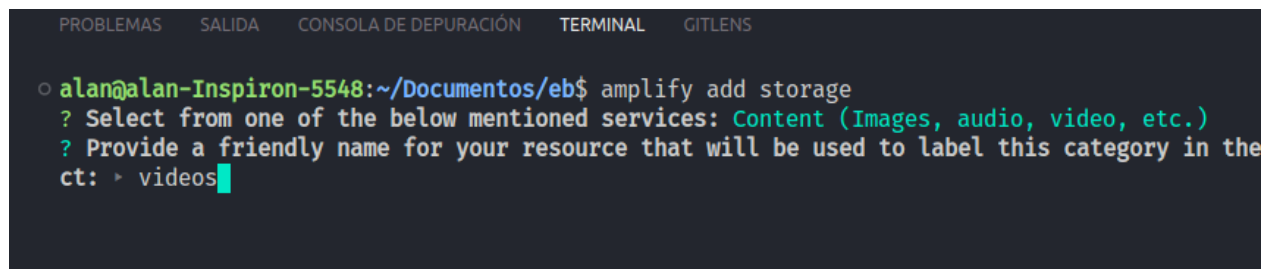
Posteriormente, se requiere especificar que tipo de servicio de almacenamiento se integrará a la aplicación (multimedia o base de datos NoSQL). Para el presente proyecto, se utilizará el almacenamiento de contenido multimedia, por lo tanto, se seleccionará dicha opción.



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  GITLENS
alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
? Select from one of the below mentioned services: (Use arrow keys)
> Content (Images, audio, video, etc.)
  NoSQL Database
```

Figura 17: Generación de Endpoint de GraphQL

Ingresamos el nombre de nuestro espacio de almacenamiento:



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  GITLENS
alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
? Select from one of the below mentioned services: Content (Images, audio, video, etc.)
? Provide a friendly name for your resource that will be used to label this category in the ct: > videos
```

Figura 18: Generación de Endpoint de GraphQL

Después, se necesita establecer cuantos usuarios, así como cuales podrán acceder a dicho servicio:

```
o alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
? Select from one of the below mentioned services: Content (Images, audio, video, etc.)
✓ Provide a friendly name for your resource that will be used to label this category in
  ct: · videos

✓ Provide bucket name: · videos
? Who should have access: ... (Use arrow keys or type to filter)
> Auth users only
  Auth and guest users
```

Figura 19: Generación de Endpoint de GraphQL

```
o alan@alan-Inspiron-5548:~/Documentos/eb$ amplify add storage
? Select from one of the below mentioned services: Content (Images, audio, video, etc.)
✓ Provide a friendly name for your resource that will be used to label this category in
  ct: · videos

✓ Provide bucket name: · videos
✓ Who should have access: · Auth and guest users
? What kind of access do you want for Authenticated users? ... (Use arrow keys or type to
  ● create/update
  ● read
  >● delete
  (Use <space> to select, <ctrl + a> to toggle all)
```

Figura 20: Generación de Endpoint de GraphQL

```
? Select from one of the below mentioned services: Content (Images, audio, video, etc.)
✓ Who should have access: · Auth and guest users
✓ What kind of access do you want for Authenticated users? · create/update, read, delete
? What kind of access do you want for Guest users? ... (Use arrow keys or type to filter)
>● create/update
  ● read
  o delete
  (Use <space> to select, <ctrl + a> to toggle all)
```

Figura 21: Generación de Endpoint de GraphQL

```

Edit your schema at /home/alan/Documentos/eb/amplify/backend/api/ebase/schema.graphql or place
graphql files in a directory at /home/alan/Documentos/eb/amplify/backend/api/ebase/schema
✓ Successfully pulled backend environment dev from the cloud.

Current Environment: dev

```

Category	Resource name	Operation	Provider plugin
Storage	videos	Create	awscloudformation
Auth	ebase35f92a6b	Update	awscloudformation
Function	S3Trigger6956e03e	No Change	awscloudformation
Api	ebase	No Change	awscloudformation

```

? Are you sure you want to continue? (Y/n) >

```

Figura 22: Generación de Endpoint de GraphQL

### 3.3. Resultados

Al ingresar a la consola de servicios de AWS, en la sección de buckets de S3, se puede observar que se encuentra el bucket recién creado llamado *videos175126-dev*.

Buckets (3) <a href="#">Info</a>				
Los buckets son contenedores de datos almacenados en S3. <a href="#">Más información</a>				
	Copiar ARN	Vaciar	Eliminar	Crear bucket
<input type="text" value="Buscar buckets por nombre"/>				
	Nombre	Región de AWS	Acceso	Fecha de creac
<input type="radio"/>	<a href="#">amplify-ebase-dev-175126-deployment</a>	EE. UU. Este (Norte de Virginia) us-east-1	Los objetos pueden ser públicos	28 Feb 2023 5
<input type="radio"/>	<a href="#">ebase6c6dcb3b214e4c3fa82df5d47dce7c22175126-dev</a>	EE. UU. Este (Norte de Virginia) us-east-1	Los objetos pueden ser públicos	25 Mar 2023 1
<input type="radio"/>	<a href="#">videos175126-dev</a>	EE. UU. Este (Norte de Virginia) us-east-1	Los objetos pueden ser públicos	17 Apr 2023 1

Figura 23: Tablas generadas mediante los schemas definidos

## 4. Creación de los servicios backend

### 4.1. Objetivo

Realizar la conexión de NodeJs con la base de datos MongoDB.

### 4.2. Descripción

Para realizar la conexión y la integración de los servicios de DynamoDB hacia nuestra API, se hará uso de AppSync. Las API de GraphQL creadas con AWS AppSync brindan a los desarrolladores frontend la capacidad de consultar varias bases de datos, microservicios y API desde un único punto de conexión de GraphQL.

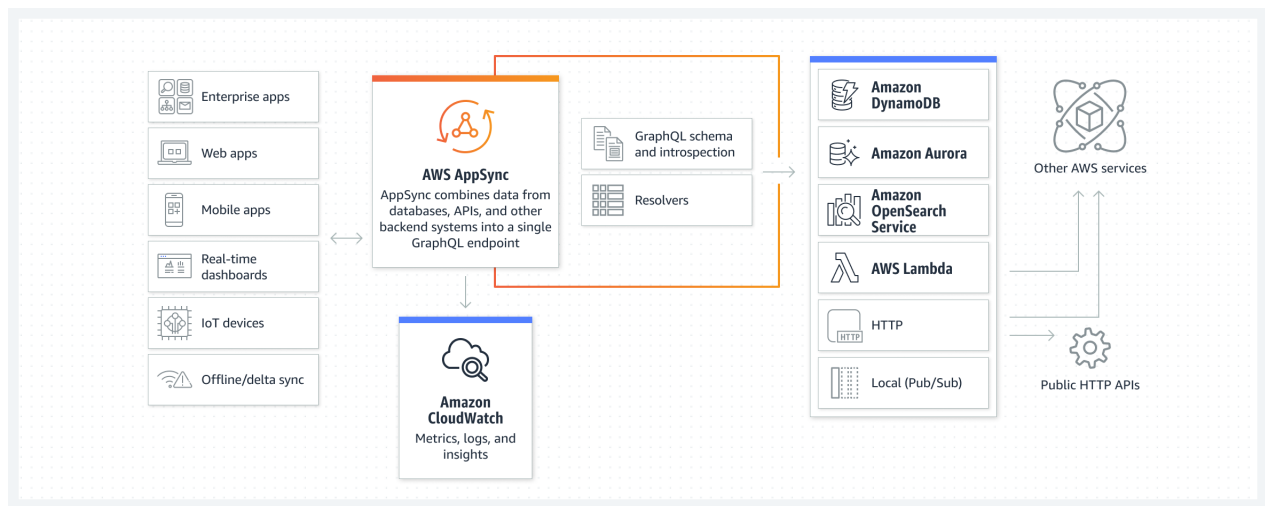


Figura 24: Funcionamiento de Appsync

AWS AppSync crea las API sin servidor de GraphQL y de publicación o suscripción que simplifican el desarrollo de aplicaciones a través de un único punto de conexión para consultar, actualizar o publicar datos.

### 4.3. Resultados

Después de haber realizado los pasos de la sección ??, si ejecutamos el comando *amplify status*, la consola de Amplify nos retornara el endpoint de GraphQL junto con AppSync correspondiente, el cuál se utilizará para realizar todas las operaciones con respecto al almacenamiento de datos



```
• alan@alan-Inspiron-5548:~/Documentos/eb$ amplify status

Current Environment: dev



| Category | Resource name | Operation | Provider plugin   |
|----------|---------------|-----------|-------------------|
| Auth     | ebase35f92a6b | No Change | awscloudformation |
| Api      | ebase         | No Change | awscloudformation |



GraphQL endpoint: https://ckqxuivcobc4rgh72skzudaixy.appsync-a
GraphQL transformer version: 2

○ alan@alan-Inspiron-5548:~/Documentos/eb$
```

Figura 25: Generación de endpoint de GraphQL

## 5. Familiarizarse con el entorno de desarrollo de la NVIDIA Jetson Nano

### 5.1. Objetivo

Investigar la documentación ofrecida por NVIDIA sobre el uso y entorno de desarrollo de la jetson nano.

### 5.2. Descripción

#### Instalación en la tarjeta microSD

El Jetson Nano Developer Kit utiliza una tarjeta microSD como dispositivo de arranque y almacenamiento principal. Por tanto fue necesario instalar un entorno de desarrollo en la propia placa, para lo cual se requirió de una tarjeta microSD de un mínimo recomendado de 32 GB de acuerdo a la documentación Nvidia. [5].

Como primer paso se descargó el *Jetson Nano Developer Kit SD Card Image* [6] y posteriormente se instaló en la tarjeta microSD desde el sistema operativo Linux, utilizando los siguientes pasos:

1. Se abrió una terminal y se insertó la tarjeta microSD.
2. Se usó el siguiente comando para mostrar que dispositivo de disco se le asignó:

```
dmesg | tail | awk '\$3 == "sd" {print}'
```

3. Se escribió la imagen de la tarjeta SD comprimida (previamente descargada) en la tarjeta microSD con el comando:

```
/usr/bin/unzip -p ~/Downloads/jetson_nano_devkit_sd_card.zip |  
sudo /bin/dd of=/dev/sda bs=1M status=progress
```

4. Finalmente se expulsó del dispositivo de disco desde la línea de comando utilizando:

```
sudo eject /dev/sda
```

#### Configuración y primer arranque

Jetson Nano Developer Kit permite dos formas de interactuar, la primera es por medio de otra computadora y la segunda haciendo uso de una pantalla, teclado y mouse conectados. Adicionalmente el kit de desarrollo no cuenta con una fuente de alimentación incluida por lo que se utilizó una fuente de alimentación Micro-USB(5V-2A).

Para iniciar el kit de desarrollo se conectó el mouse, la pantalla, el teclado y la fuente de alimentación, posteriormente se realizó la configuración inicial del sistema operativo el cual incluyó lo siguiente:

- Revisar y aceptar el EULA del software NVIDIA Jetson.
- Seleccionar el idioma del sistema, la distribución del teclado y la zona horaria
- Crear nombre de usuario, contraseña y nombre de la computadora.
- Seleccione el tamaño de partición de la aplicación: se recomienda utilizar el tamaño máximo sugerido.

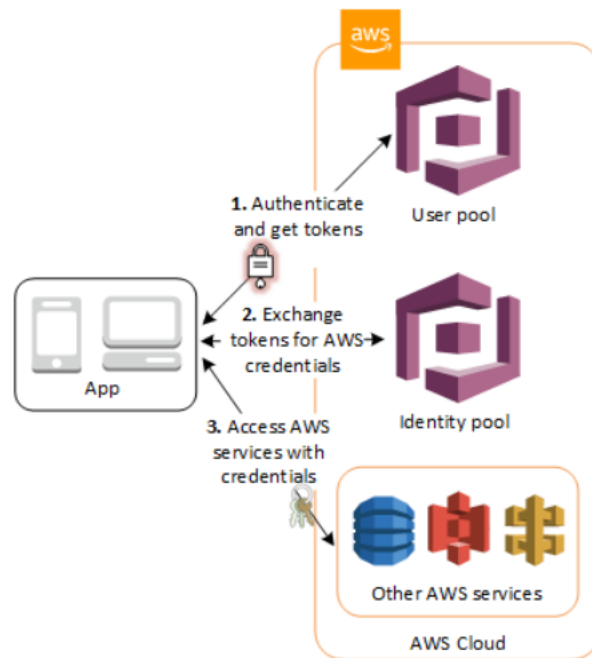


Figura 26: Funcionamiento de Amazon Cognito

### 5.3. Resultados

## **6. Implementar algoritmo para la detección del rostro y ojos**

### **6.1. Objetivo**

realizar la detección de rostro y ojos en la Jetson Nano .

### **6.2. Descripción**

### **6.3. Resultados**

## **7. Implementar puntos faciales en el rostro y la metrica MOR**

### **7.1. Objetivo**

Implementar la detección con puntos faciales en el rostro y la metrica mor para detectar si la boca se encuentra abierta o cerrada.

### **7.2. Descripción**

### **7.3. Resultados**

## 8. Conclusiones

## 9. Bibliografía

### Referencias

- [1] F. Martinez. *Protección de rutas con React Router Dom*, DEV Community. <https://dev.to/franklin030601/proteccion-de-rutas-con-react-router-dom-144j> (accedido el 7 de marzo de 2023).
- [2] *¿Qué es Amazon DynamoDB?*, Amazon Docs. <https://docs.aws.amazon.com/es-es/amazondynamodb/latest/developerguide/Introduction.html> (accedido el 2 de marzo de 2023).
- [3] *API sin servidor de GraphQL y de publicación o suscripción – AWS AppSync – Amazon Web Services*. Amazon Web Services, Inc. <https://aws.amazon.com/es/appsync/> (accedido el 12 de marzo de 2023).
- [4] *AWS — Gestión de identidades y autenticación de usuario en la nube*, Amazon Web Services, Inc. <https://aws.amazon.com/es/cognito/> (accedido el 8 de marzo de 2023).
- [5] NVIDIA, "Get Started with the Jetson Nano Developer Kit", NVIDIA Developer, 2019. [Online]. Disponible: <https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#intro>. [Accedido: Abril 02 2023].
- [6] Nvidia Developer, "Get Started with Jetson Nano Devkit," Nvidia Developer. [En línea]. Disponible: <https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#write>. [Accedido: 2 de abril de 2023].
- [7] Dusty, N. "Building the Repo - NVIDIA Jetson Inference," GitHub. [Online]. Disponible en: <https://github.com/dusty-nv/jetson-inference/blob/master/docs/building-repo-2.md>. [Accedido en: 02-abr-2023].