

# Manual de Instalación de Servicios en AWS para el Agente de Respuesta SIGOAVE

Arroyo Auz Christian Xavier

## 1. Introducción

El agente de respuesta forma parte de un conjunto de sistemas autónomos para la gestión de accidentabilidad vehicular, este es una aplicación móvil que presenta el índice de accidentabilidad del vehículo y ciertos parámetros claves del conductor, vehículo, condiciones climatológicas y del flujo de tráfico. Adicionalmente, este agente se encarga de solicitar información procesada al servicio de siniestrabilidad vehicular (SIGOAVE).

Además, la aplicación genera notificaciones y alertas en base a la información provista por SIGOAVE. Una de las principales características de esta aplicación móvil es que funcione de manera autónoma, pero pueda trabajar en sinergia con la aplicación móvil del agente de adquisición.

La Figura 1 presenta el esquema de este conjunto de sistemas autónomos.

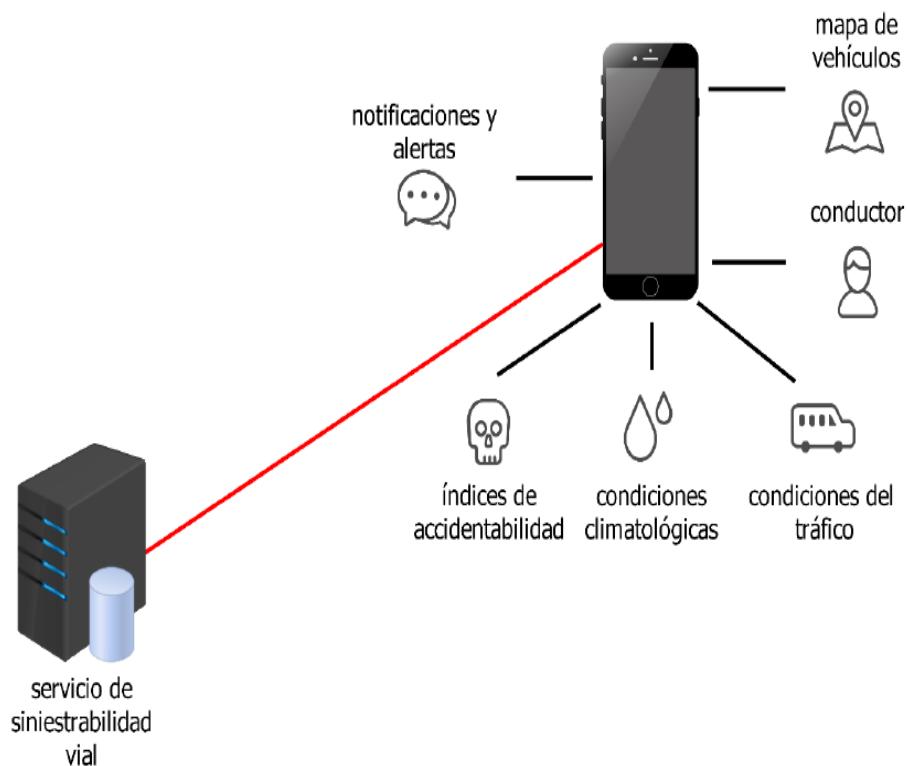


Figura 1: Funcionalidades del Agente de Respuesta.

En la era digital actual, la nube se ha convertido en el pilar fundamental para el desarrollo y despliegue de aplicaciones y servicios. Amazon Web Services (AWS) se destaca como líder en el ámbito de servicios en la nube, ofreciendo una amplia gama de soluciones para empresas y desarrolladores. Este documento se centrará en el proceso de creación de servicios en AWS, para el Agente de Respuesta SIGOAVE, explorando los pasos esenciales para aprovechar al máximo la plataforma y proporcionar soluciones eficientes y escalables.

La creación de servicios en AWS representa una respuesta a la creciente demanda de flexibilidad y eficiencia en el desarrollo de aplicaciones móviles y la gestión de infraestructuras. AWS ofrece una gama diversa de servicios, desde servicios de cómputo y almacenamiento hasta herramientas de inteligencia artificial y aprendizaje automático, permitiendo a los desarrolladores y empresas construir soluciones personalizadas y altamente escalables.

## 2. Objetivo

El objetivo principal al crear servicios en AWS es aprovechar la flexibilidad y la escalabilidad que la nube ofrece, permitiendo adaptarse rápidamente a las demandas cambiantes del mercado. Los servicios en la nube proporcionan una infraestructura sólida y confiable, reduciendo la carga operativa y permitiendo a los equipos enfocarse en la innovación en lugar de en la gestión de la infraestructura física.

El proyecto incluye el diseño, desarrollo e implementación de un conjunto de agentes para gestión de accidentabilidad en vehículos. Los agentes propuestos pretenden mejorar los índices de accidentabilidad, disminuir los tiempos de respuesta para intervención y contribuir a evitar accidentes de tránsito. Los agentes poseerán autonomía de manera que podrán actuar en forma individual y conjunta por medio de mecanismos de retroalimentación para conseguir sistemas coordinados y actualizados. Estos medirán la probabilidad o grado de ocurrencia que un vehículo sufra un accidente de tránsito, y en base a esa información tomarán decisiones que disminuyan la probabilidad de que el accidente tome lugar. El sistema está formado por los agentes de adquisición, procesamiento y respuesta.

### 2.1. Objetivos Específicos

- Agente de adquisición: es una aplicación móvil que recolectará información de un lector OBD, un receptor GPS, y otros sensores, y enviará información hacia un repositorio de datos en Internet.
- Agente de procesamiento: es una herramienta que usará computación en Internet, técnicas de minería de datos y algoritmos de aprendizaje de máquina para realizar la clasificación y procesamiento de la información almacenada en el repositorio de datos. El modelo de aprendizaje medirá la probabilidad o grado de ocurrencia que un vehículo sufra un accidente de tránsito.
- Agente de respuesta: es una aplicación móvil que permitirá a usuarios disponer de resultados (niveles de accidentabilidad), estadísticas y notificaciones.

## 3. Esquema del Modelo de Agente

Se diseñó e implementó una aplicación móvil para Android basada en el lenguaje de programación Java. La aplicación interactúa con un escáner OBD2 y

un receptor GPS para recoger parámetros y la ubicación del vehículo, y con un reloj inteligente para obtener el ritmo cardíaco del conductor. Una vez que la información es generada, se realiza la obtención de los parámetros de condiciones climáticas desde un servicio climatológico y el número de accidentes de tránsito dados cerca de una ubicación específica desde una base de datos. En primera instancia, el agente de adquisición fue usado para generar el conjunto de datos de conducción denominada POLIDriving. Esta versión del conjunto de datos incluye información del conductor, del vehículo, condiciones climáticas y accidentes de tránsito. Una vez que el conjunto de datos fue generado, el agente de adquisición tiene la funcionalidad de recoger los parámetros y almacenarlos localmente. El agente de respuesta usará esta información para enviar al modelo y realizar predicciones. La Figura 2 presenta el esquema de funcionamiento del agente de adquisición.

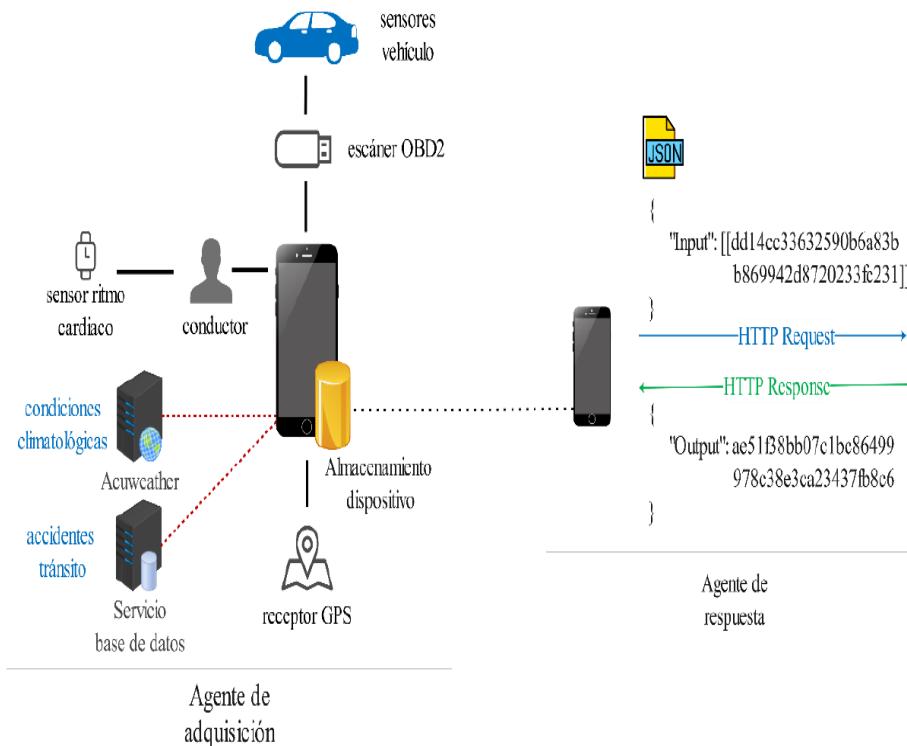


Figura 2: Esquema de Funcionamiento del Agente de Adquisición.

El artículo [Marcillo et al., 2022] y publicado en Applied Sciences (Q2), proporcionó información relacionada a fuentes de datos y atributos usados por los modelos de predicción, y a plataformas, servicios de Internet y simuladores usados para recolectar información. Esta información permitió diseñar e implementar un agente de adquisición que se ajuste a ciertos requerimientos funcionales y no funcionales. En el siguiente enlace se encuentra el conjunto de datos POLIDriving:

- [https://github.com/laboratorioAI/PIS\\_20\\_02\\_AGENTES\\_IA/tree/main/Actualizacion\\_2024/Dataset](https://github.com/laboratorioAI/PIS_20_02_AGENTES_IA/tree/main/Actualizacion_2024/Dataset)

## 4. Implementación Servicios AWS

El agente de procesamiento consta de 2 fases:

- Creación, entrenamiento e implementación del modelo de aprendizaje de máquina.
- Configuración y despliegue del API REST para consultar en tiempo real el nivel de riesgo de sufrir un accidente de tránsito.

Para la primera fase se creó, entrenó, e implementó un modelo usando el servicio SageMaker de Amazon Web Services (AWS). Mientras que para la segunda fase se configuró y desplegó un API REST usando los servicios SageMaker, Lambda, API Gateway y S3 de AWS. La Figura 3 presenta el esquema de infraestructura usado por el agente de procesamiento.

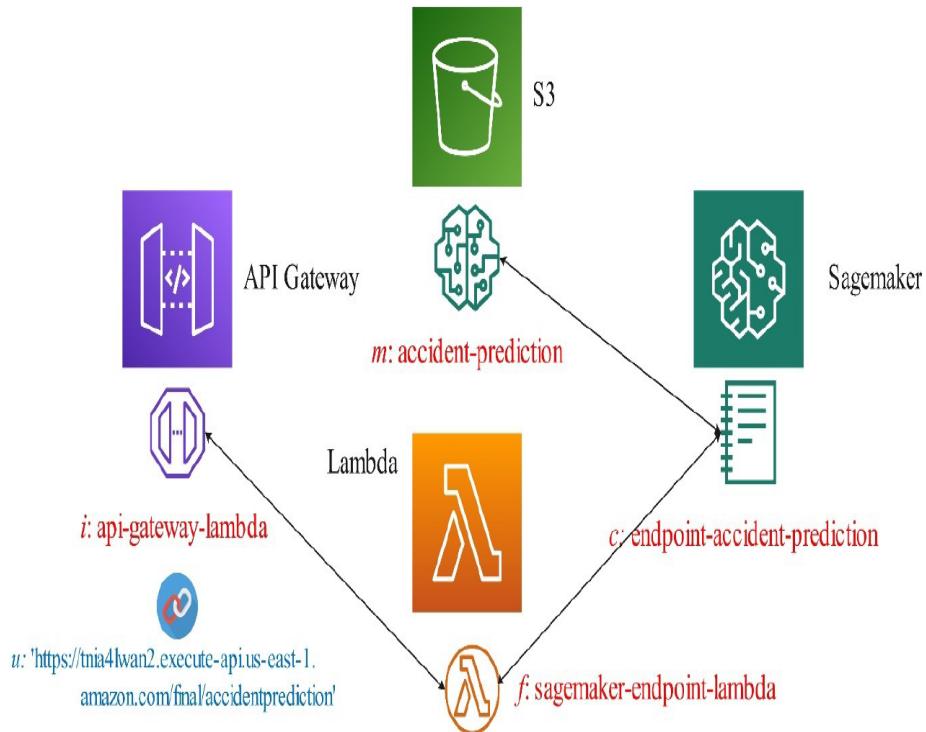


Figura 3: Infraestructura del agente de procesamiento.

### 4.1. Administración de Identidades - AWS IAM

El primer paso para la creación de los servicios en AWS, es la creación de la “Identidad de Usuario”, mediante el uso del Administrador de Identidades. Con AWS Identity and Access Management (IAM), puede especificar quién o qué puede acceder a los servicios y recursos en AWS, administrar de forma centralizada los permisos específicos y analizar el acceso para perfeccionar los permisos en todo AWS [AWS, 2023d]. La Figura 4 muestra el funcionamiento de IAM.

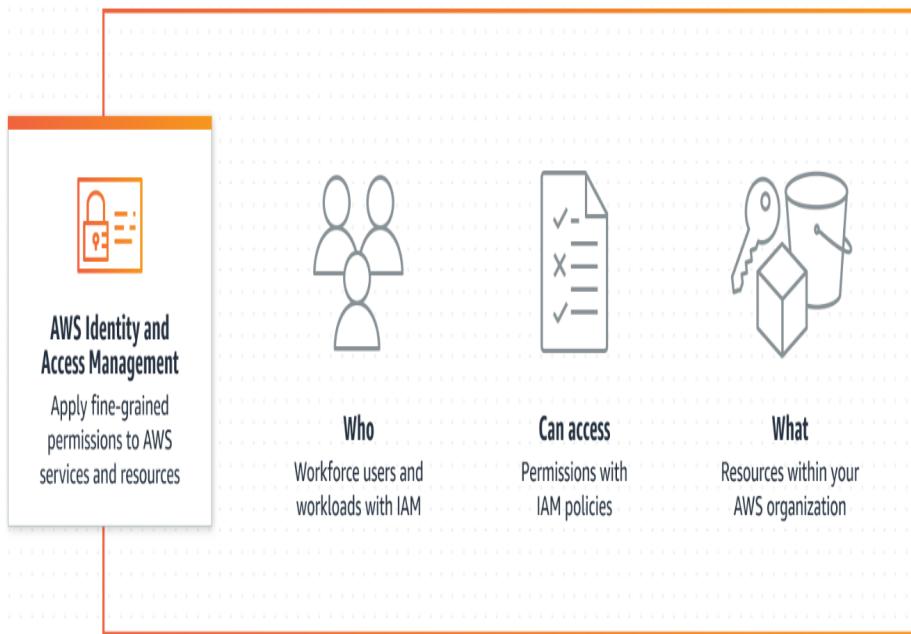


Figura 4: Cómo funciona la gestión de acceso e identidad de AWS.

Para la creación de una nueva identidad es necesario dirigirse a la consola de AWS y buscar el servicio de IAM como muestra la Figura 5, una vez localizado el servicio se ingresa al mismo y se puede apreciar el panel de control de IAM.

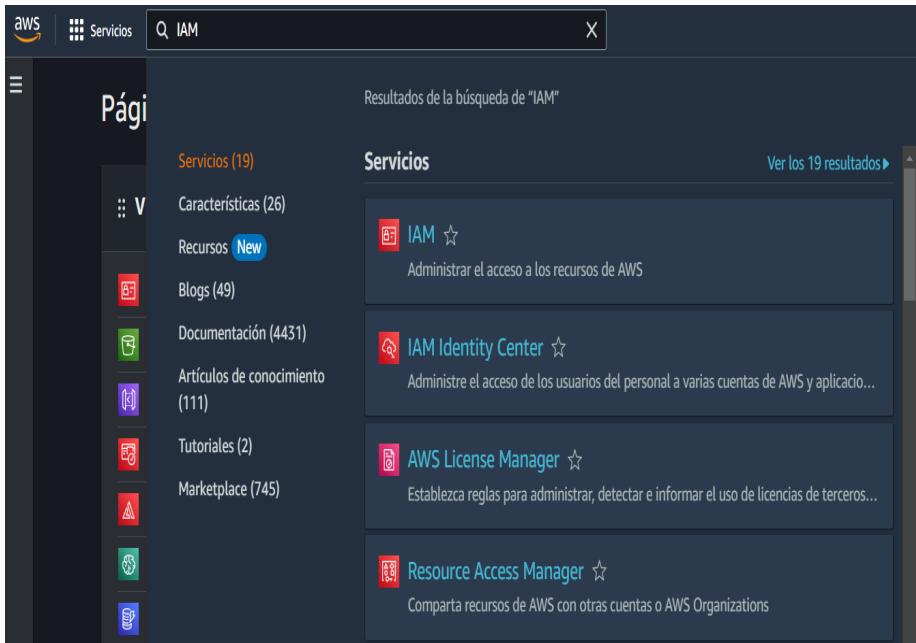


Figura 5: Búsqueda Servicio IAM.

En el panel de control de IAM se localiza la opción “Usuarios” como muestra la Figura 6. Se ingresa a este dando “Clic” sobre el número, en un principio este sera 0, esto lleva a una nueva pantalla en la que se podrá crear un nuevo usuario como muestra la Figura 7.

The screenshot shows the AWS IAM Panel. On the left, there's a sidebar with navigation options like 'Panel', 'Grupos de usuarios', 'Usuarios', 'Roles', 'Políticas', 'Proveedores de identidad', and 'Configuración de cuenta'. Below that, under 'Informes de acceso', are 'Access Analyzer', 'Acceso externo', 'Acceso no utilizado', and 'Configuración del analizador'. The main area is titled 'Panel de IAM' and contains two sections: 'Recomendaciones de seguridad' and 'Recursos de IAM'. In the 'Recursos de IAM' section, there are five categories: 'Grupos de usuarios' (0), 'Usuarios' (2, highlighted with a red box), 'Roles' (24), 'Políticas' (2), and 'Proveedores de identidad' (0). A button 'Agregar MFA' is visible in the 'Recomendaciones de seguridad' section.

Figura 6: Usuarios IAM.

The screenshot shows the 'Users' page within the IAM service. The sidebar on the left is identical to Figure 6. The main area is titled 'Usuarios (2) Información'. It displays a table with one row for 'app\_pi2002'. The columns include 'Nombre de usuario' (app\_pi2002), 'Ruta' (/), 'Grupo' (0), 'Última actividad' (hace 1 hora), 'MFA' (-), 'Antigüedad de' (-), and 'Último inicio de sesión' (-). There are buttons for 'Crear usuario' (yellow) and 'Eliminar' (grey).

Figura 7: Crear Usuario IAM.

En la siguiente pantalla se debe ingresar un nombre de usuario, este nombre debe ser único y reconocible; posteriormente, se debe dar “Clic” en el botón “Siguiente” como muestra la Figura 8. Esto redirigirá a una nueva pantalla donde se podrá escoger las políticas que se desee dar al usuario, para ello se escoge la opción “Adjuntar políticas directamente”. Podemos escoger cuantas políticas deseemos, ver Figura 9. En futuro se pueden agregar más políticas de ser necesario.

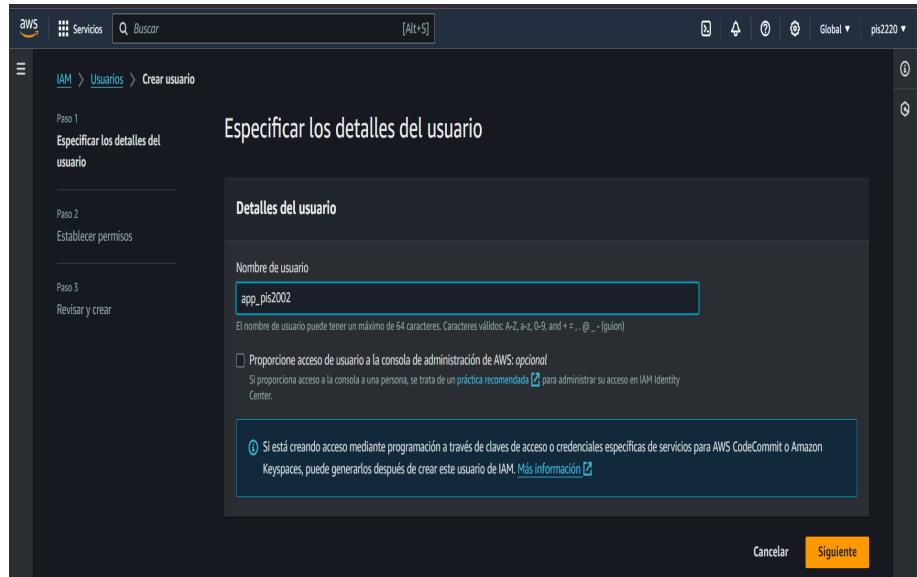


Figura 8: Nombre de Usuario IAM.

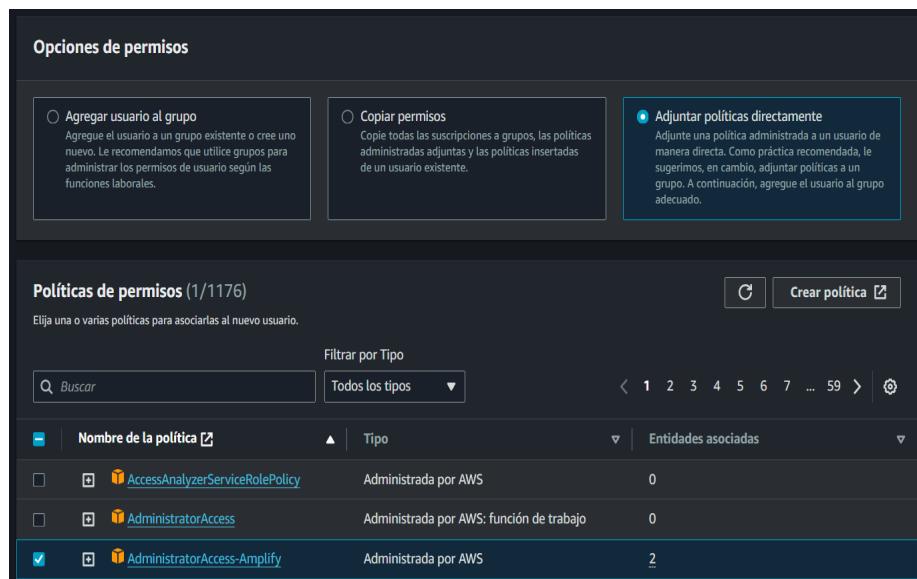


Figura 9: Permisos Usuario IAM.

Tras agregar las políticas deseadas al usuario, se pasa a una nueva interfaz donde se muestra un resumen de las políticas tomadas para el usuario. Ver Figura 10.

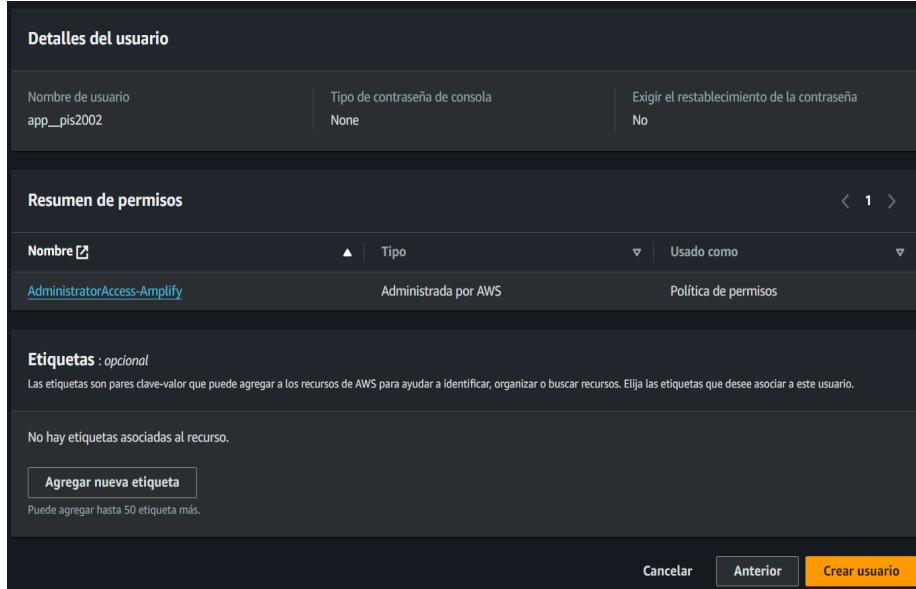


Figura 10: Resumen Permisos IAM.

Finalmente, se crea el usuario presionando en el botón “Crear Usuario”, tras lo cual se redirigirá a la consola de IAM. En la consola se puede apreciar que el usuario a sido creado con éxito. Ver Figura 11.

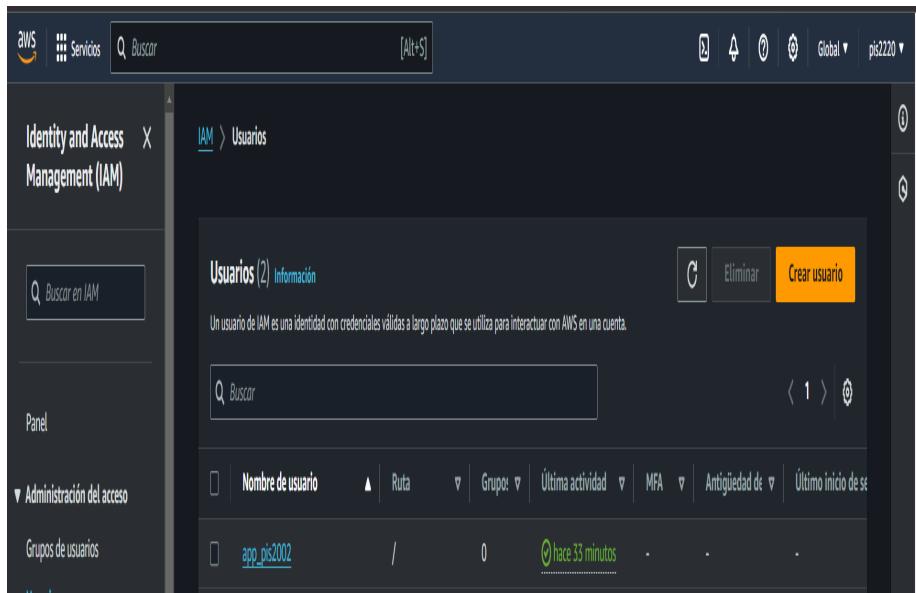


Figura 11: Usuario IAM Creado.

Creado el usuario, ingresamos al mismo dando “Clic” sobre el nombre del usuario y se muestra un resumen detallado de las características del usuario, ver Figura 12. En esta página se tiene que seleccionar la opción “Credenciales de seguridad”, dentro de “Credenciales de seguridad” se busca la opción “Claves de acceso” y se crea una nueva clave de acceso dando “Clic” sobre el botón “Crear clave de acceso”, Ver Figura 13

The screenshot shows the AWS Identity and Access Management (IAM) service interface. On the left, there's a sidebar with navigation options like 'Panel', 'Administración del acceso' (selected), 'Grupos de usuarios', 'Usuarios' (highlighted in blue), 'Roles', 'Políticas', 'Proveedores de identidad', and 'Configuración de cuenta'. The main content area is titled 'Identity and Access Management (IAM)' and shows a user named '9user/app\_pi52002'. It displays details such as ARN, Access to the console status (Desactivada), and two access keys: 'Clave de acceso 1' (Active, last used today) and 'Clave de acceso 2' (with a 'Create key' link). Below this, there's a section for 'Permisos' (Permissions) with a sub-section for 'Políticas de permisos (4)'. A red box highlights the 'Credenciales de seguridad' tab. At the bottom, there's a large 'Crear clave de acceso' (Create new key) button.

Figura 12: Detalle Usuario IAM.

This screenshot shows the 'Claves de acceso' (Access Keys) creation page. It has a header with 'Claves de acceso (0)' and a 'Crear clave de acceso' (Create new key) button. Below this, there's a note about using access keys for long-term calls and a link to 'Más información'. The main message states 'No hay claves de acceso. Como práctica recomendada, evite el uso de credenciales a largo plazo, como las claves de acceso. En su lugar, utilice herramientas que proporcionen credenciales a corto plazo.' (There are no access keys. As a recommended practice, avoid using long-term credentials like access keys. Instead, use tools that provide short-term credentials.) Another 'Crear clave de acceso' button is located at the bottom.

Figura 13: Clave de Acceso IAM.

En la siguiente ventana se selecciona “Interfaz de línea de comandos (CLI)” y “Entiendo la recomendación anterior y deseo proceder a la creación de una clave de acceso” como se aprecia en la Figura 14.



Figura 14: Casos de uso IAM.

Agregamos una descripción a la clave y posteriormente se muestra la clave creada. En la Figura 15 se muestra la clave de acceso creada, en ella se aprecia que se crearon dos claves: “Clave de acceso” y “Clave de acceso secreta”. Se debe recordar almacenar esta información en un lugar seguro ya que en futuro no se podrá acceder a esta información.

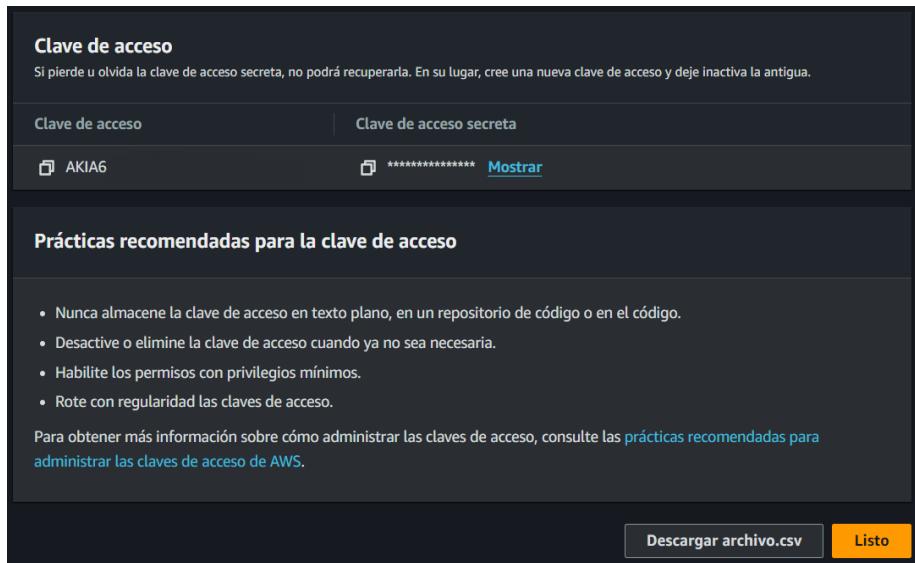


Figura 15: Access y Secret Key IAM.

Finalmente, al presionar “Listo” en la pantalla anterior se regresa a la pantalla principal. En esta pantalla se muestra un resumen detallado de las nuevas “Clases de acceso” y ver si esta activa o no. Ver Figura 16.

The screenshot shows the AWS IAM Access Keys page with one item listed:

Clave de acceso (1)	Crear clave de acceso
AKIA6 Descripción: proyecto Último uso: Ninguno Última región utilizada: N/A	Estado: Active Creado: hace 1 minuto Último servicio utilizado: N/A

Figura 16: Detalles de Claves de Acceso.

#### 4.2. Machine Learning Service - AWS SageMaker

El siguiente paso es la implementación del modelo en “SageMaker” de AWS. Amazon SageMaker es un servicio de aprendizaje automático totalmente gestionado. SageMaker permite a los desarrolladores y a los analistas de datos crear y perfeccionar modelos de aprendizaje automático de forma rápida y sencilla y, a continuación, implementarlos directamente en un entorno alojado listo para su uso [AWS, 2023e]. En la Figura 17 se aprecia el funcionamiento de SageMaker.

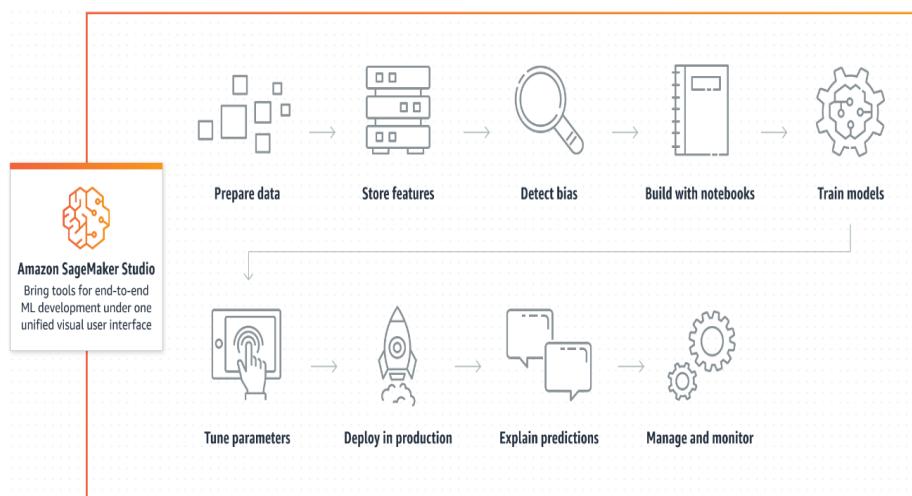


Figura 17: Funcionamiento de SageMaker.

Antes de empezar con la configuración de SageMaker con el usuario con el cual se va a trabajar es necesario agregar una nueva política llamada “AdministratorAccess”, para ello como se menciona en el apartado 4.1 en los detalles del usuario se puede agregar nuevas políticas, ver Figura 18.

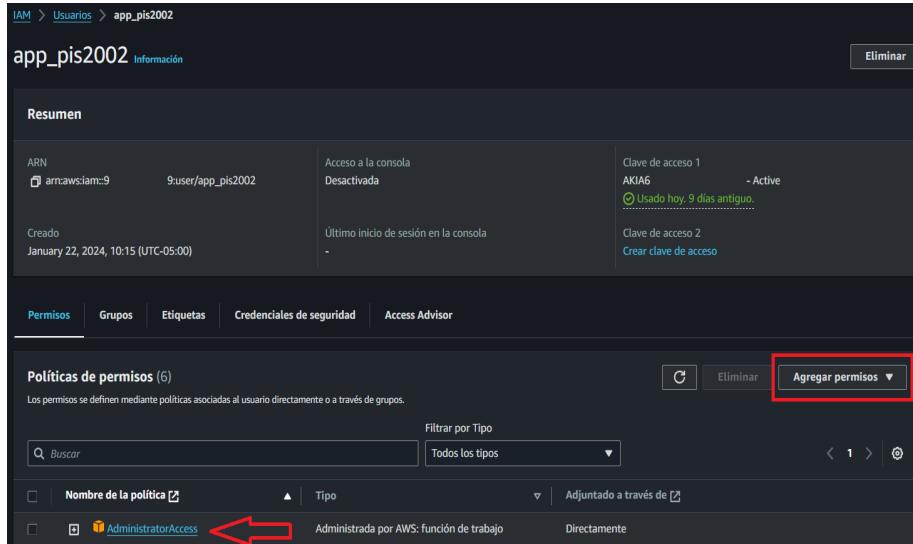


Figura 18: Agregación Nueva Política.

Adicionalmente, es necesario tener un “DataSet” con extensión “CSV” con el cual se va a trabajar. Este archivo se encuentra en la dirección URL mencionada en el apartado 3, de los diferentes archivos de datos que se encuentran se usar DataSet cuyo etiqueta es “20221215\_151443\_pre”, en la Figura 19 se muestra una parte del contenido del mencionado DataSet.

	A	B	C	D	F	F	G	H	I	J	K	L	M	N	O	P
1	Id,time,steering_angle,speed,rpm,acceleration,throttle_position,engine_temperature,system_voltage,barometric_pressure,distance_travelled,distance_travelled_total,id_vehicle,latitude,longitude,altitude,id_driver,heart_rate,street_name															
2	0,15:14:56,38,7,0,663,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330874831,-78,43229134,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,38,7,4,663,0,1,3,1,2															
3	1,15:14:57,38,7,0,665,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330874831,-78,43229134,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,38,7,4,665,0,1,3,1,2															
4	2,15:14:58,35,8,0,668,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330874831,-78,43229134,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,8,4,668,0,1,3,1,2															
5	3,15:14:59,35,5,0,670,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330874831,-78,43229134,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,670,0,1,3,1,2															
6	4,15:15:00,35,5,0,664,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330874831,-78,43229134,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,664,0,1,3,1,2															
7	5,15:15:01,35,5,0,673,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330874831,-78,43229134,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,673,0,1,3,1,2															
8	6,15:15:02,35,5,0,668,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330874831,-78,43229134,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,668,0,1,3,1,2															
9	7,15:15:03,35,5,0,665,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330864564,-78,43229955,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,665,0,1,3,1,2															
10	8,15:15:04,35,5,0,681,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330844633,-78,4322993,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,681,0,1,3,1,2															
11	9,15:15:05,35,5,0,672,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330863977,-78,4322979,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,672,0,1,3,1,2															
12	10,15:15:06,35,5,0,665,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,330863977,-78,4322979,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,665,0,1,3,1,2															
13	11,15:15:07,35,5,0,669,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,669,0,1,3,1,2															
14	12,15:15:08,35,5,0,672,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,672,0,1,3,1,2															
15	13,15:15:09,35,5,0,670,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,670,0,1,3,1,2															
16	14,15:15:10,35,5,0,669,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,669,0,1,3,1,2															
17	15,15:15:11,35,5,0,668,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,668,0,1,3,1,2															
18	16,15:15:12,35,7,0,671,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,7,4,671,0,1,3,1,2															
19	17,15:15:13,35,9,0,670,0,12,94117647,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,66,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,9,4,670,0,1,3,1,2															
20	18,15:15:14,38,6,0,677,0,12,94117647,89,12,7,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,72,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,38,6,4,677,0,1,3,1,2															
21	19,15:15:15,35,5,0,645,0,21,56862745,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,72,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,35,5,4,645,0,1,3,1,2															
22	20,15:15:16,64,4,0,1565,0,17,64705882,89,12,8,10,58775487,0,0,2,141879087,1,-0,33087634,-78,43234791,0,2,72,0,0,2,Mostly cloudy,0,0,1,24,6,21,9,29,9,0,59,16,1,2,Low,86,4389,1015,9,0,5,21,64,4,4,1565,0,1,3,1,2															

Figura 19: Parte de Contenido DataSet.

Para la creación de este servicio es necesario dirigirse a la consola de AWS y buscar el servicio de SageMaker como muestra la Figura ??, una vez localizado el servicio se ingresa al mismo y se puede apreciar el panel de control. En el panel de control localizamos la opción “Bloc de Notas” y dentro de esta opción localizamos “Instancias de bloc de notas” como muestra la Figura 20.

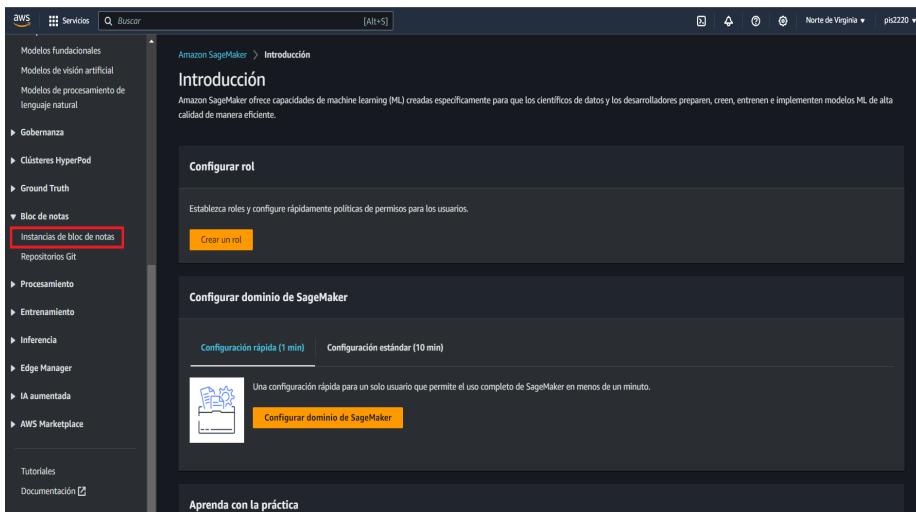


Figura 20: Búsqueda NoteBook de SageMaker.

Al ingresar al panel de “Instancias de bloc de notas” se puede crear una nueva instancia tan solo con dar “Clic” en el botón de “Crear instancia de bloc de notas” como muestra la Figura 21. Al hacer esto redirigirá a un nuevo formulario.

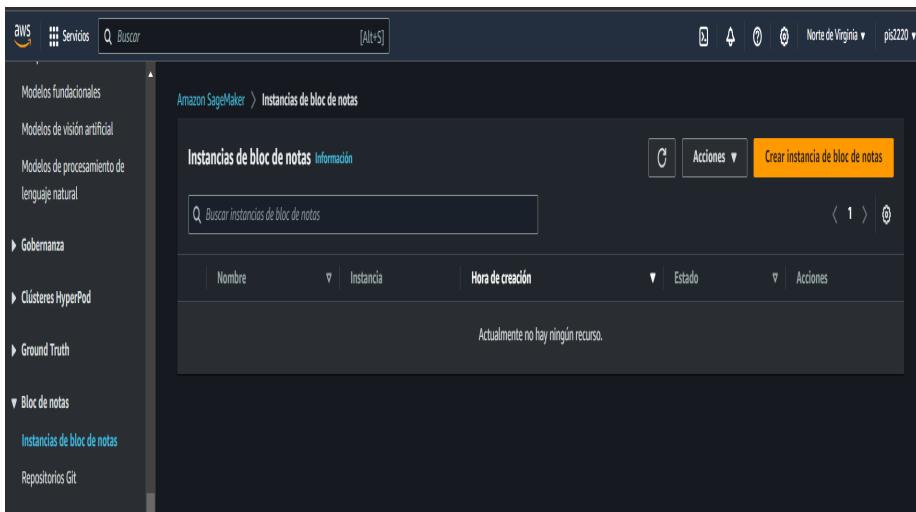


Figura 21: Crear NoteBook de SageMaker.

En la página “Crear instancia de bloc de notas”, en el cuadro de Configuración de la instancia de bloc de notas, se rellena los siguientes campos:

- El Nombre de la instancia.
- En Tipo de instancia: ml.t2.medium.
- En Inferencia elástica, mantener la selección predeterminada.
- En Identificador de plataforma, mantener la selección predeterminada.

La Figura 22 muestra las diferentes configuraciones realizadas.

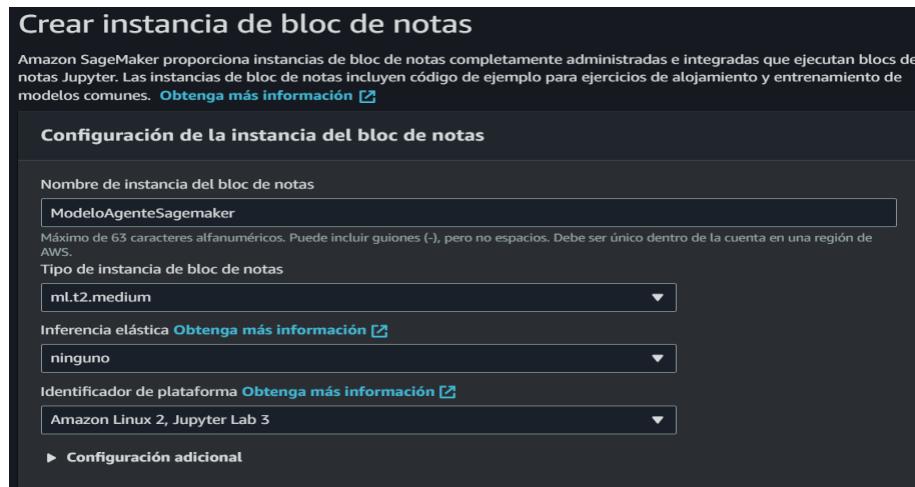


Figura 22: Configuración de la instancia del bloc de notas.

Una vez realizada la configuración de la instancia del bloc de notas, se procede a configurar los “Permisos y cifrado”. En la sección Permisos y cifrado, para el rol de IAM, se selecciona “Crear un nuevo rol” como muestra la Figura ?? y, en el cuadro de diálogo “Crear una función de IAM”, seleccione “Cualquier bucket S3” y seleccionar “Crear rol” como muestra la Figura 23.

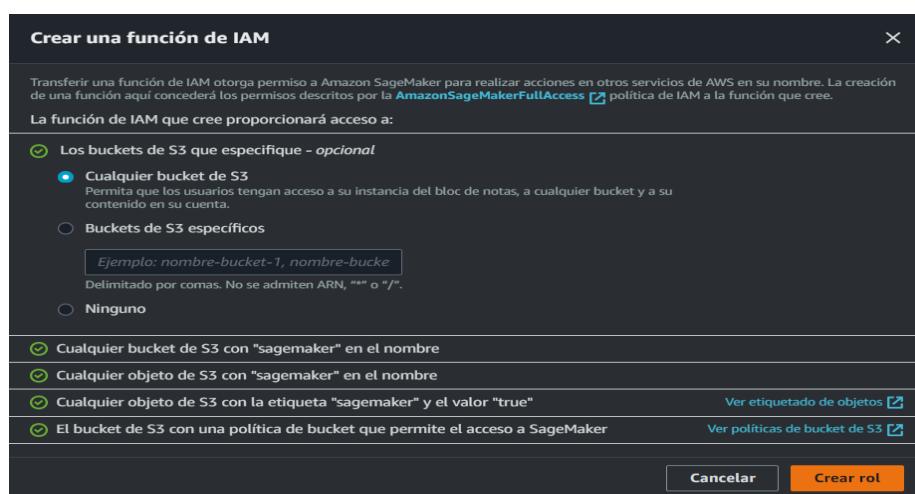


Figura 23: Crear una función de IAM.

Amazon SageMaker creará el rol AmazonSageMaker-ExecutionRole. Ver Figura 24.

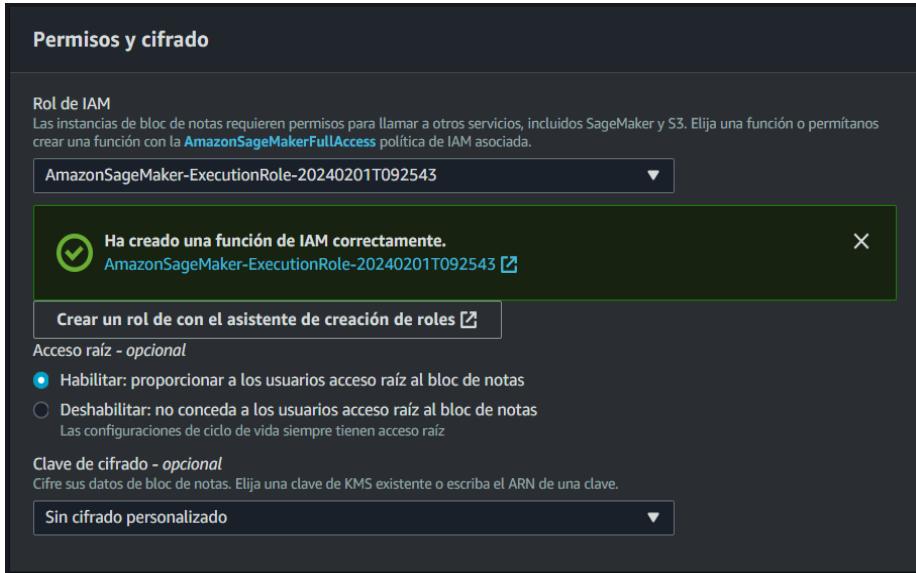


Figura 24: SageMaker ExecutionRole.

Conservar la configuración predeterminada para las demás opciones y elegir “Crear instancia de bloc de notas”. En la sección Instancias de bloc de notas, se muestra la instancia de cuaderno recién creada con un estado “Pendiente”, ver Figura 25. El cuaderno estará listo cuando el estado cambie a “En Servicio” lo cual puede tardar varios minutos, ver Figura 26.

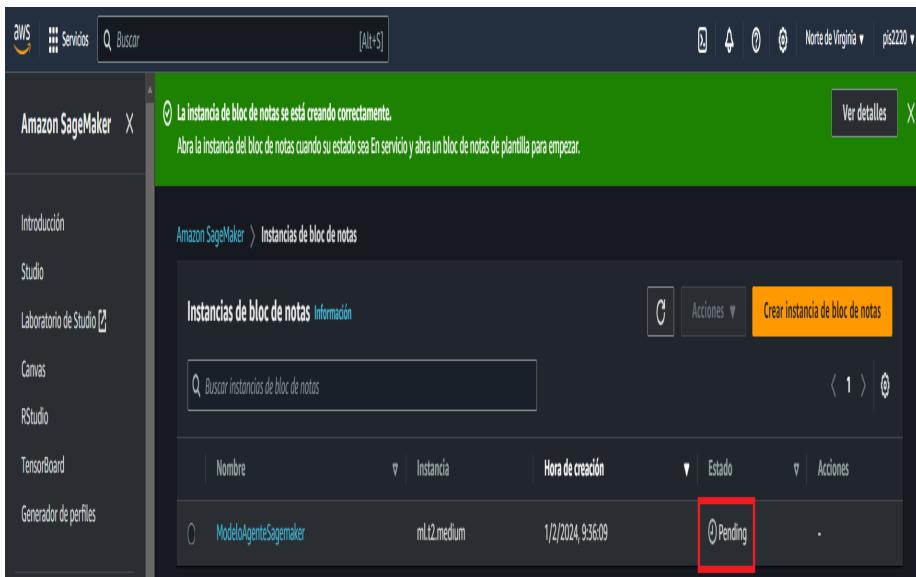


Figura 25: Instancia Creada SageMaker.

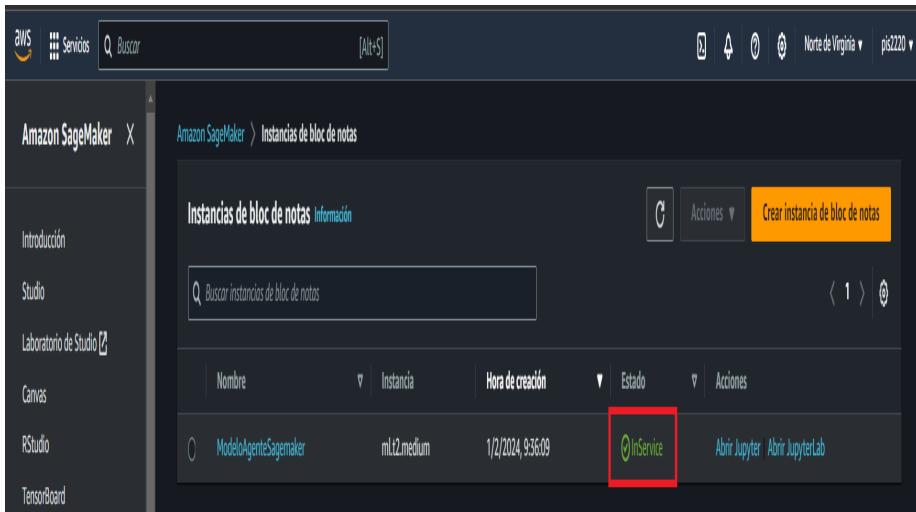


Figura 26: Instancia Creada SageMaker en Servicio.

En este paso, se utiliza la instancia de cuaderno de Amazon SageMaker para pre-procesar los datos que se necesita para entrenar el modelo de machine learning y, a continuación, cargar los datos en Amazon S3.

Después de que el estado de la instancia de cuaderno cambie a “En Servicio”, seleccionar “Abrir Jupyter”, como muestra la Figura 27.

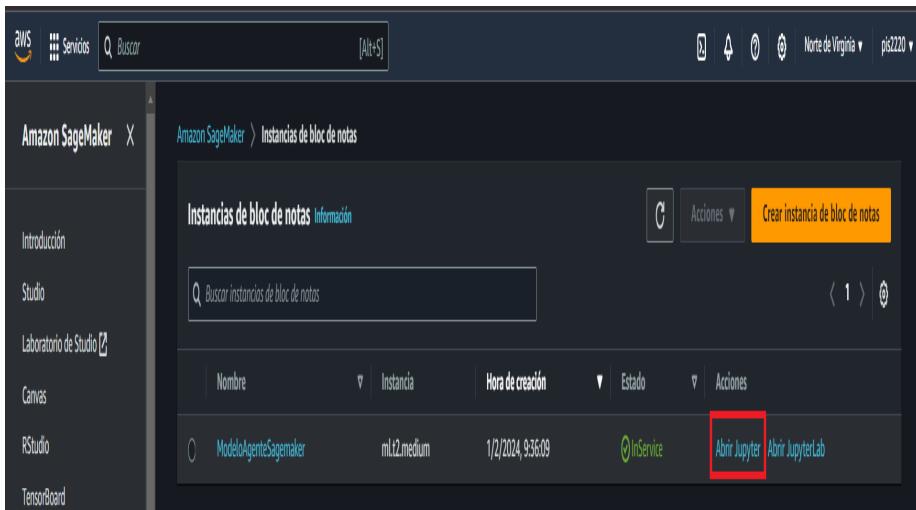


Figura 27: Abrir Jupyter.

JupyterLab es el último entorno de desarrollo interactivo basado en web para cuadernos, códigos y datos. Su interfaz flexible permite a los usuarios configurar y organizar flujos de trabajo en ciencia de datos, informática científica, periodismo computacional y aprendizaje automático. Un diseño modular invita a ampliaciones para ampliar y enriquecer la funcionalidad. [JupyterLab, 2024]

En el entorno de Jupyter, seleccionar “Cargar” para poder subir el DataSet con el cual se va a trabajar, a seleccionar esta opción se abrirá un cuadro de dialogo que permitirá buscar en el equipo el DataSet y subirlo al entorno de Jupyter, ver Figura 28. Una vez cargado se mostrara en la interfaz principal el DataSet, ver Figura 29.

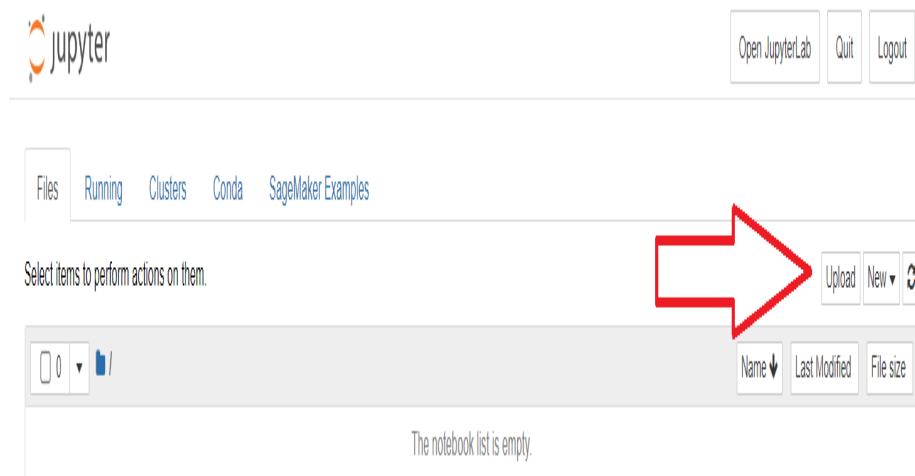


Figura 28: Jupyter DataSet.

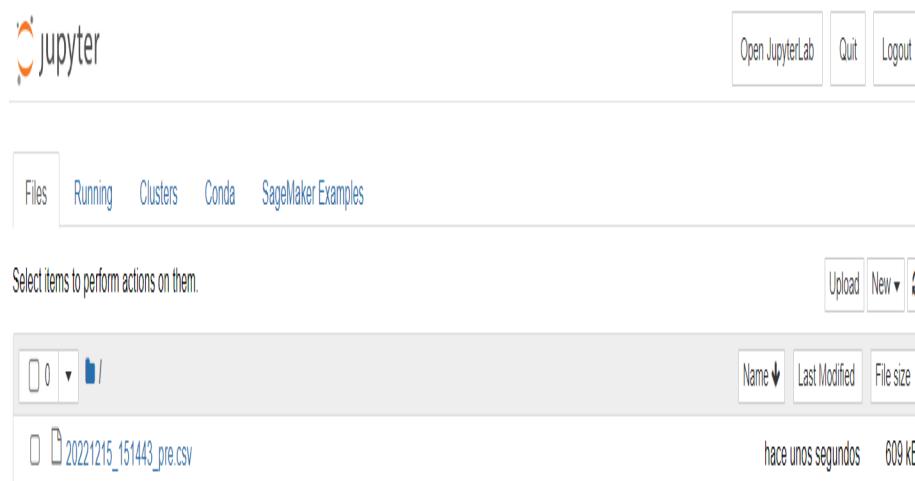


Figura 29: Jupyter DataSet Cargado.

En el entorno de Jupyter, seleccionar “Nuevo” y a continuación, seleccionar “conda\\_python3” como muestra la Figura 30. Esto creara un nuevo NoteBook de Python, en el cual ya se puede comenzar a utilizar código de Python para crear el agente. En la sección de “Archivo”, “Guardar como” se le puede cambiar el nombre al documento. En este caso se le pondrá el nombre de “Modelo”, ver Figura 31.

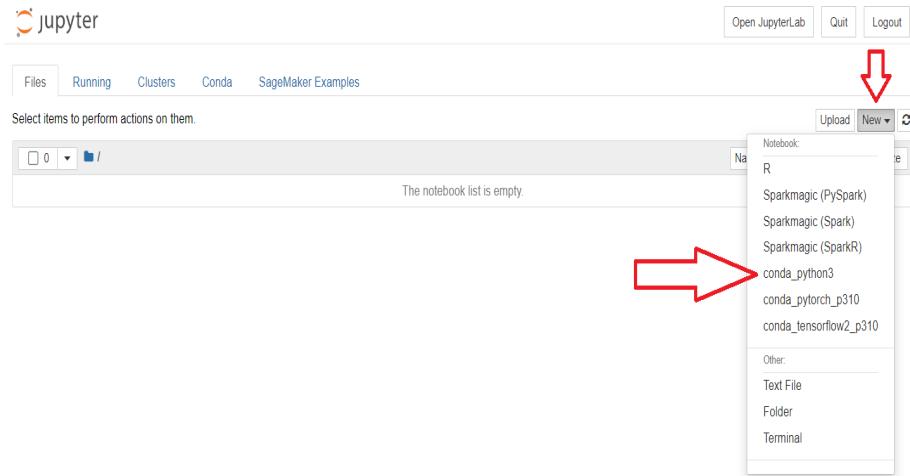


Figura 30: Entorno de Jupyter.

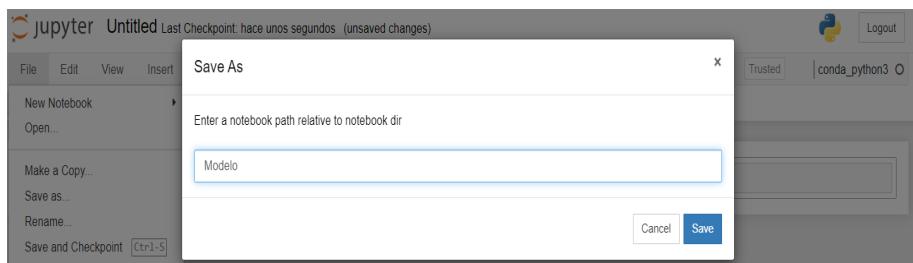


Figura 31: Cambio Nombre del Agente.

Una vez subido el DataSet y configurado el nombre se puede comenzar a realizar código. El primer paso es crear un archivo con extensión “py”, para ello seleccionar “Nuevo” y a continuación, seleccionar “Archivo de Texto” como muestra la Figura 32

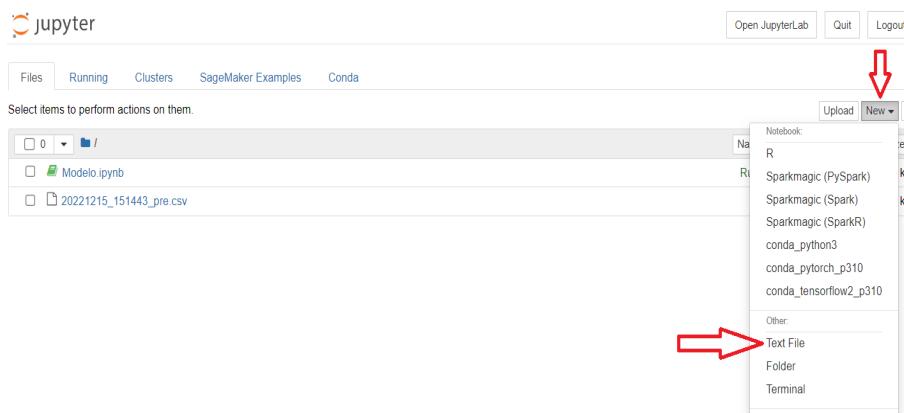
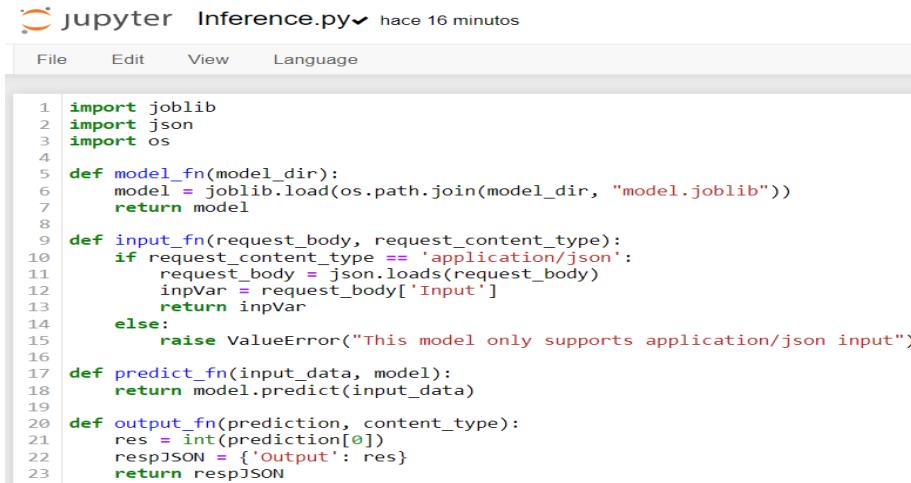


Figura 32: Nuevo Archivo de Texto.

Al igual que en el caso anterior mostrado en la Figura 31, se puede cambiar el nombre del archivo. En este caso se llamará al archivo “Inference.py”, no olvidar poner la extensión “.py” en el nombre del documento. Una vez que se cambió el nombre se puede proceder a ingresar código en el interior del archivo, la Figura 33 muestra un segmento de código del archivo “Inference.py”.



```

 1 import joblib
 2 import json
 3 import os
 4
 5 def model_fn(model_dir):
 6     model = joblib.load(os.path.join(model_dir, "model.joblib"))
 7     return model
 8
 9 def input_fn(request_body, request_content_type):
10     if request_content_type == 'application/json':
11         request_body = json.loads(request_body)
12         inpVar = request_body['Input']
13         return inpVar
14     else:
15         raise ValueError("This model only supports application/json input")
16
17 def predict_fn(input_data, model):
18     return model.predict(input_data)
19
20 def output_fn(prediction, content_type):
21     res = int(prediction[0])
22     respJSON = {'Output': res}
23     return respJSON

```

Figura 33: Código de Inference.py.

En el archivo “Modelo.ipynb” creado anteriormente se programa el agente como tal, la Figura 34 muestra una parte del código que se implementó.



```

In [1]: # Carga de las librerías necesarias para el entrenamiento del modelo
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sagemaker import image_uris
import pandas as pd
import numpy as np
import subprocess
import sagemaker
import warnings
import joblib
import boto3

sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.config/sagemaker/config.yaml

In [2]: # Carga del conjunto de datos y elección de las características
filename_dataset = '20221215_151443_pre.csv'
df = pd.read_csv(filename_dataset)
X = df[['steering_angle', 'speed', 'rpm', 'acceleration', 'throttle_position', 'engine_temperature',
        'system_voltage', 'heart_rate', 'distance_travelled', 'latitude', 'longitude', 'current_weather',
        'accidents_onsite']]
y = df[['accident_rate']]
X = X.dropna(axis = 0)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20)

In [3]: # Configurar y entrenar el modelo
mlp = MLPClassifier(hidden_layer_sizes = (10, 10, 10), max_iter = 1000)
mlp.fit(X_train, y_train.values.ravel())
predictions = mlp.predict(X_test)

In [4]: # Exportación del modelo
with open('model.joblib', 'wb') as f:
    joblib.dump(mlp, f)

In [5]: # Comprimiendo el Modelo
command = "tar -cvzf model.tar.gz model.joblib Inference.py"
process = subprocess.Popen(command.split(), stdout=subprocess.PIPE)
output, error = process.communicate()

```

Figura 34: Código de Modelo.ipynb.

Una vez que se ejecutaron todas la líneas de código en el NoteBook se puede observar varios resultados, los cuales se muestran a continuación:

- En JupyterLab se crearon varios archivos, ver Figura 35.
- Se crea un nuevo Bucket de uso general, ver Figura 36.
- Se crea un nuevo Modelo de Inferencia, ver Figura 37.
- Se crea un nuevo punto de conexión, ver Figura 38.
- Se crea un nuevo punto final, ver Figura 39.

The screenshot shows the JupyterLab interface with the following details:

- Header:** jupyter, Open JupyterLab, Quit, Logout
- Navigation:** Files, Running, Clusters, SageMaker Examples, Conda
- Toolbar:** Upload, New, Select items to perform actions on them.
- File List:**

	Name	Last Modified	File size
Modelo.ipynb	Running hace 15 minutos	6.42 kB	
20221215_151443_pre.csv	hace una hora	609 kB	
Inference.py	hace una hora	635 B	
model.joblib	hace 15 minutos	17.2 kB	
model.tar.gz	hace 15 minutos	15 kB	

Figura 35: Jupyter Archivos.

The screenshot shows the AWS S3 console with the following details:

- Header:** Amazon S3, Instantánea de la cuenta, Ver panel de Storage Lens
- Navigation:** Buckets de uso general, Buckets de directorio
- Buckets de uso general:**
  - Nombre:** sagemaker-us-east-1-975050286309
  - Región de AWS:** EE. UU. Este (Norte de Virginia) us-east-1
  - Acceso:** Bucket y objetos que no son públicos
  - Fecha de creación:** 1 Feb 2024 11:56:15 AM -05
- Buttons:** C, Copiar ARN, Vaciar, Eliminar, Crear bucket
- Search Bar:** Buscar buckets por nombre
- Pagination:** < 1 >

Figura 36: S3 SageMaker Bucket.

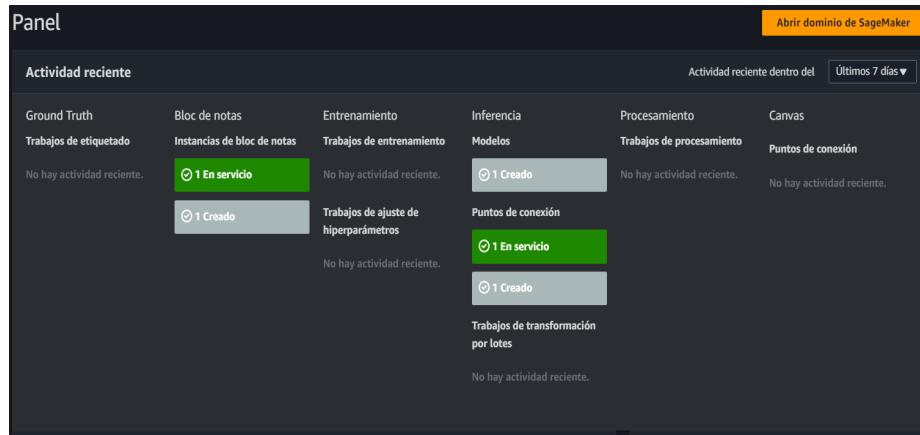


Figura 37: SageMaker Inferencia.

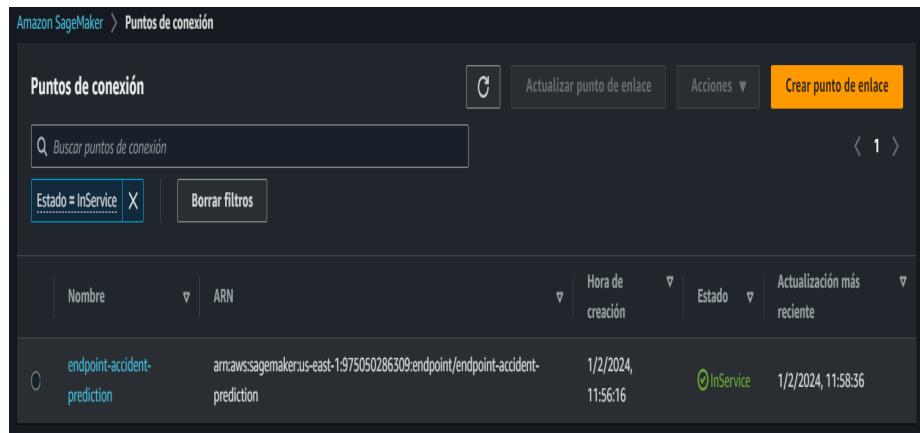


Figura 38: SageMaker Punto de Conexión.



Figura 39: SageMaker EndPoint.

En el siguiente enlace se encuentra el código para generar y desplegar el modelo de predicción y la configuración para crear el punto final en el servicio Sagemaker:

- [https://github.com/laboratorioAI/PIS\\_20\\_02\\_AGENTES\\_IA/tree/main/Actualizacion\\_2024/Modelo](https://github.com/laboratorioAI/PIS_20_02_AGENTES_IA/tree/main/Actualizacion_2024/Modelo)

#### 4.2.1. Gestión de recursos informáticos - AWS Lambda

En este paso comprende la “Gestión de recursos informáticos” mediante AWS Lambda. AWS Lambda es un servicio informático sin servidor y basado en eventos que le permite ejecutar código para prácticamente cualquier tipo de aplicación o servicio backend sin necesidad de aprovisionar o administrar servidores [AWS, 2023f]. En la Figura 40 se aprecia el funcionamiento de Lambda.



Figura 40: Funcionamiento de Lambda.

Para incorporar este servicio es necesario ir a la consola de AWS y buscar el servicio de Lambda como muestra la Figura 41.

AWS Services Search: Lambda

Resultados de la búsqueda de "Lambda"

Servicios (7)

- Características (3)
- Recursos New
- Blogs (27)
- Documentación (2235)
- Artículos de conocimiento (160)
- Tutoriales (4)
- Marketplace (747)

Servicios

- Lambda ☆ Ejecute código sin tener que pensar en los servidores
- CodeBuild ☆ Compile y pruebe código
- AWS Signer ☆ Garantizar la confianza y la integridad del código
- Amazon Inspector ☆ Administración continua de las vulnerabilidades a escala

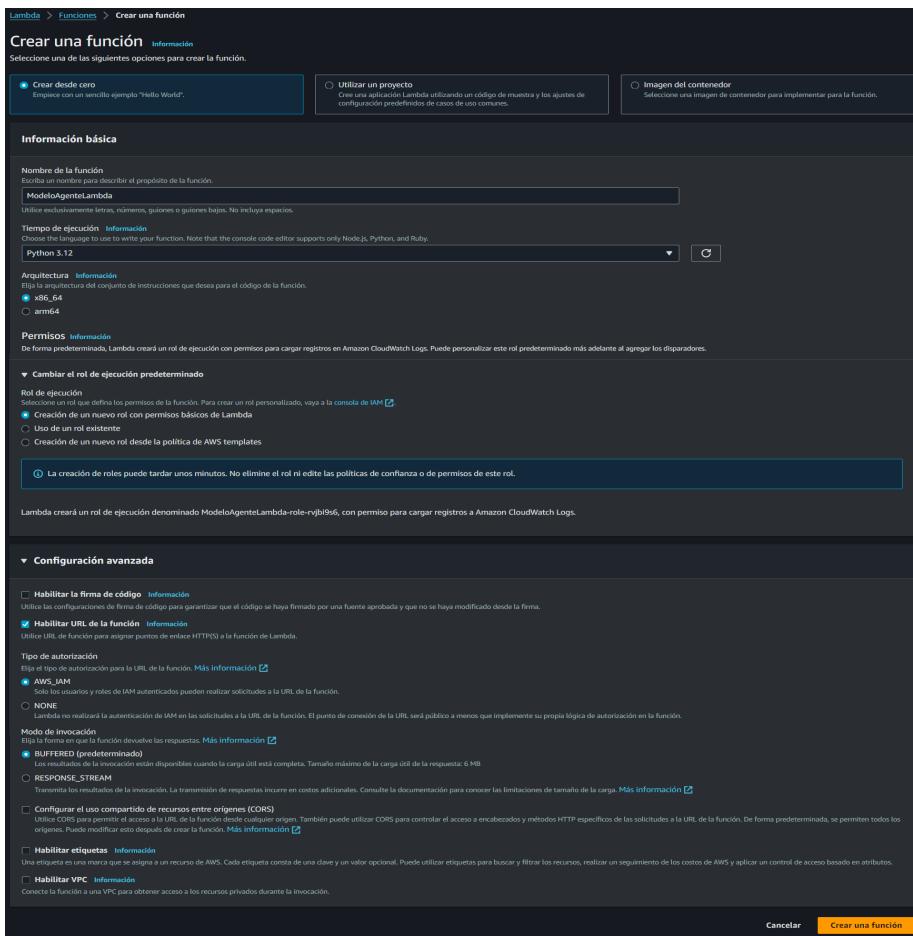
Figura 41: Búsqueda Servicio Lambda.

Una vez dentro del panel de control de Lambda se debe crear una nueva función de Lambda dando “Clic” en el botón de “Crear una función”, ver Figura 42.

Recursos para EE.UU. Este (Norte de Virginia)			
Función(es) de Lambda	Almacenamiento de código	Simultaneidad de cuenta completa	Simultaneidad de cuenta no reservada
6	13,0 kB (0 % de 75,0 GB)	10	10

Figura 42: Crear Función Lambda.

Una vez dentro de la consola de “Crear funciones Lambda” es necesario configurar la función, ver Figura 43 de la siguiente manera: Tipo de función: Crear desde cero, El nombre de la función,Tiempo de ejecución: Python 3.12, Arquitectura: x86\_64, Rol de ejecución: Creación de un nuevo rol con permisos básicos de Lambda,Configuración avanzada: Habilitar URL de la función.



The screenshot shows the 'Create a function' step in the Lambda console. It includes sections for basic information (function name: 'ModeloAgenteLambda', runtime: 'Python 3.12', architecture: 'x86\_64'), permissions (choose a role), and advanced configuration (enable HTTP triggers). A note at the bottom states: 'Lambda will create a new execution role named ModeloAgenteLambda-role-rvjbis6, with permission to log to Amazon CloudWatch Logs.'

Figura 43: Configuración Lambda.

Continuando con la configuración, en la pantalla mostrada en la Figura 44, se muestra el detalle de la función creada en Lambda. En este detalle se puede apreciar el diagrama dela función, el ARN de la función y la dirección URL de la función.

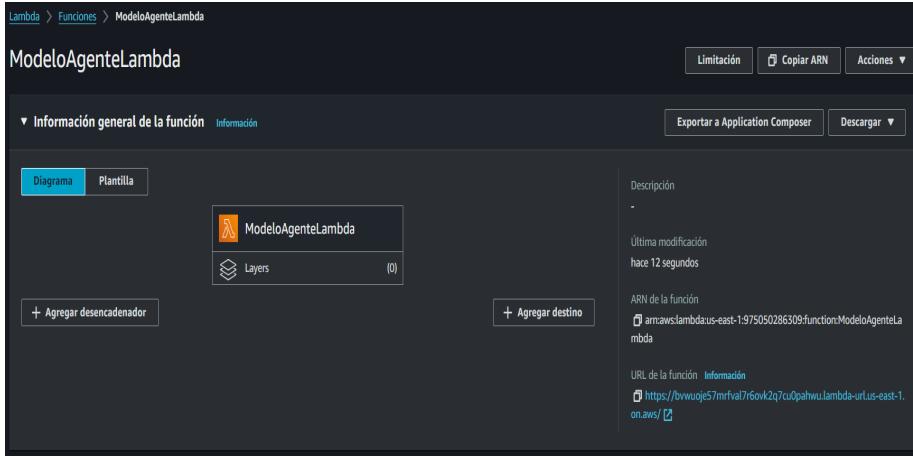


Figura 44: Detalle Configuración Lambda.

Dentro de la misma pantalla en la parte baja, se puede encontrar la sección de “Configuración” y en ella la sección de “Permisos”, donde se puede ver el “Nombre del rol” creado. Ver Figura 45. Si se hace “Clic” en el nombre del rol llevará a la consola de IAM y se podrá ver mas detalles sobre el rol.

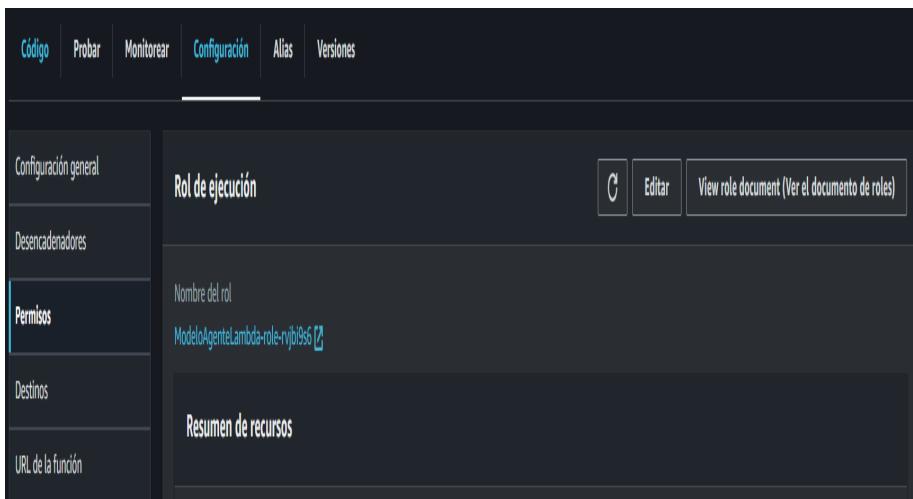


Figura 45: Nombre de Rol Lambda.

En la nueva pantalla de debe crear una nueva política. Ya sea que haya creado una nueva función o se utilice una función existente, hay que asegúrese de incluir la siguiente política, que otorga a la función permiso para invocar un punto final del modelo, para ello se selecciona el nombre de la política como muestra la Figura 46.

The screenshot shows the AWS IAM Roles interface. In the top navigation bar, 'Roles' is selected under 'ModeloAgenteLambda-role-rvjb9s6'. The main section is titled 'Resumen' (Summary). It displays basic information: 'Fecha de creación' (February 01, 2024, 15:41 (UTC-05:00)), 'ARN' (arn:aws:iam::975050286309:role/service-role/ModeloAgenteLambda-role-rvjb9s6), 'Última actividad' (None), and 'Duración máxima de la sesión' (1 hora). Below this, the 'Permisos' tab is active, showing 'Políticas de permisos (1/1)'. A single policy, 'AWSLambdaBasicExecutionRole-25c11397-1b06-4...', is listed. At the bottom right of this section, there is a red box around the 'Agregar permisos' (Add permissions) button.

Figura 46: Crear política insertada.

En la nueva pantalla de “Permisos definidos en esta política” se debe escoger la opción “JSON”, ver Figura 47, y posteriormente seleccionar la opción “Editar” y agregar la política manualmente como muestra la Figura 48. Finalmente, se procede a finalizar con la creación de la política. Ver Figura 49.

The screenshot shows the 'Permissions defined in this policy' page. At the top, there are tabs: 'Permisos' (Permissions), 'Entidades asociadas' (Associated entities), 'Etiquetas' (Tags), 'Versiones de la política (1)' (Policy versions), and 'Access Advisor'. The 'JSON' tab is highlighted with a red box and an arrow pointing to it from the left. Below this, a message states: 'Los permisos definidos en este documento de política especifican qué acciones se permiten o deniegan. Para definir permisos para una identidad de IAM (usuario, grupo de usuarios o rol), asóciela a una política'. A search bar is present. The main content area shows a table with one row:

Servicio	Nivel de acceso	Recurso	Condición de solicitud
CloudWatch Logs	Limitado: Escribir	Múltiple	None

Figura 47: Selección JSON.

```

1  {
2    "Version": "2012-10-17",
3    "Statement": [
4      {
5        "Effect": "Allow",
6        "Action": "logs:CreateLogGroup",
7        "Resource": "arn:aws:logs:us-east-1:975050286309:log-group:/aws/lambda/ModeloAgentelambda"
8      },
9      {
10        "Effect": "Allow",
11        "Action": [
12          "logs:CreateLogStream",
13          "logs:PutLogEvents"
14        ],
15        "Resource": [
16          "arn:aws:logs:us-east-1:975050286309:log-group:/aws/lambda/ModeloAgentelambda"
17        ]
18      },
19      {
20        "Sid": "VisualEditor0",
21        "Effect": "Allow",
22        "Action": "sagemaker:InvokeEndpoint",
23        "Resource": "*"
24      }
25    ]
26  }

```

Figura 48: Política insertada JSON.

Servicio	Nivel de acceso	Recurso	Condición de solicitud
CloudWatch Logs	Limitado: Escribir	Múltiple	None
SageMaker	Limitado: Leer	Todos los recursos	None

Establezca esta nueva versión como predeterminada.  
Los permisos definidos en esta versión se aplicarán a todas las entidades a las que está adjunta la política.

Figura 49: Creación total política insertada.

Tras finalizar se puede apreciar en la pantalla principal que la nueva política ha sido creada con éxito, ver Figura 50. esta política se creó automáticamente con el nombre de “SageMaker”.

Servicio	Nivel de acceso	Recurso	Condición de solicitud
CloudWatch Logs	Limitado: Escribir	Múltiple	None
SageMaker	Limitado: Leer	Todos los recursos	None

Figura 50: Resumen Políticas Lambda.

Regresando al panel de control de Lambda se procede a programar el código fuente de la función Lambda, en la Figura 51 se muestra el código fuente antes de desplegar la función Lambda.

```

1 import boto3
2 import json
3 import os
4 import io
5
6 ENDPOINT_NAME = os.environ['ENDPOINT_NAME']
7 runtime = boto3.client('runtime.sagemaker')
8
9 def lambda_handler(event, context):
10     data = json.loads(json.dumps(event))
11     response = runtime.invoke_endpoint(EndpointName = ENDPOINT_NAME, Body = json.dumps(data))
12     result = json.loads(response['Body'].read().decode())
13
    return result

```

Figura 51: Código Fuente Lambda.

Luego de ingresar el código fuente, en la misma pantalla se ingresa en la opción “Configuración” y luego en “Variables de entorno” seleccionamos la opción “Editar”, ver Figura 52. Esto redirigirá al usuario a una nueva consola donde deberá configurar las Variables de entorno.

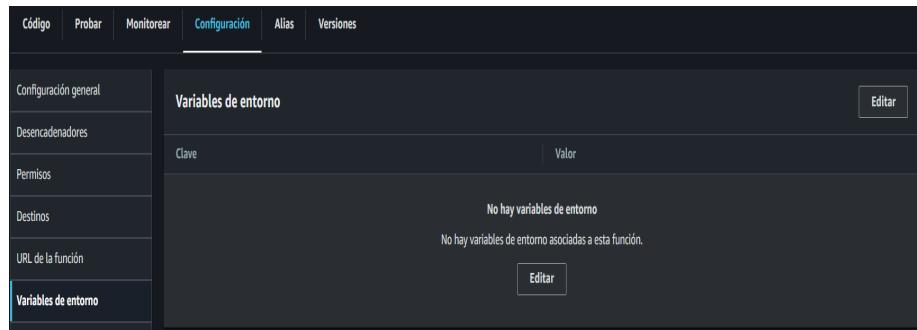


Figura 52: Variables de Entorno.

En la pantalla de configuración de variables se debe escoger la opción “Agregar variable de entorno” y agregar la “Clave” y el “Valor” como muestra la Figura 53. En el campo de Clave se debe poner la siguiente información: “ENDPOINT\_NAME”, mientras que en el campos de valor se debe poner la información que se presenta en la Figura 38.



Figura 53: Configuración de Variable de Entorno.

Tras guardar la configuración realizada se puede observar que en el panel de funciones de Lambda en la sección Configuración y Variables de entorno se a creado con éxito el “EndPoint” de Lambda, ver Figura 54.

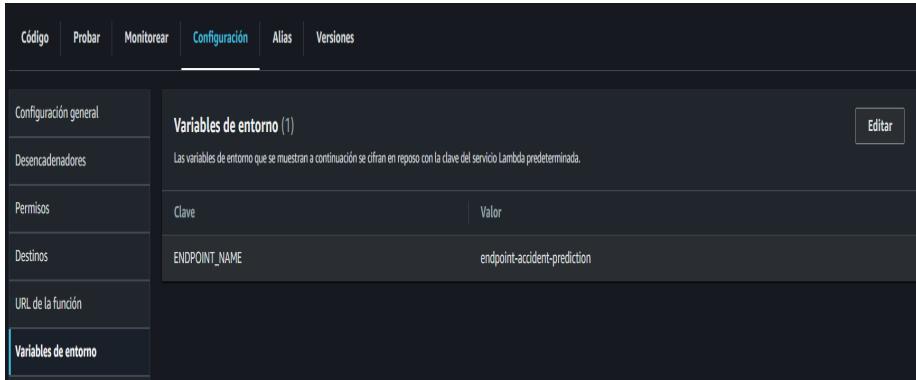


Figura 54: EndPoint Creado.

Finalmente, en la misma pantalla se regresa a la sección de “Código” y se presiona el botón de “Deploy” para desplegar la función Lambda como muestra la Figura 55.

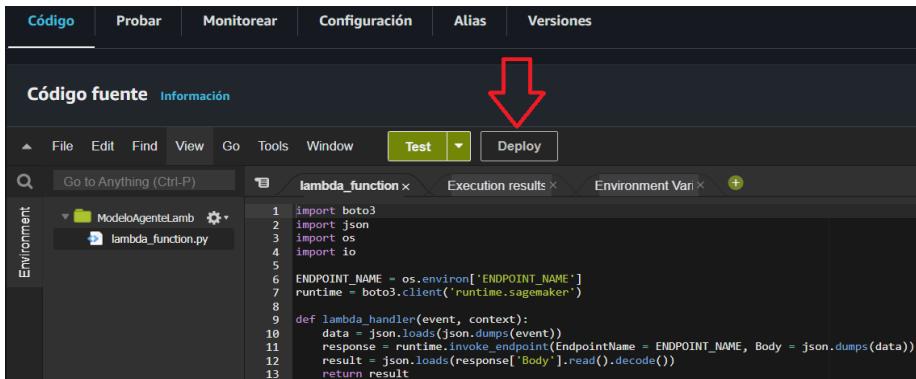


Figura 55: Deploy Función Lambda.

En el siguiente enlace se encuentra el código de implementación de la función Lambda:

- [https://github.com/laboratorioAI/PIS\\_20\\_02\\_AGENTES\\_IA/tree/main/Actualizacion\\_2024/Lambda](https://github.com/laboratorioAI/PIS_20_02_AGENTES_IA/tree/main/Actualizacion_2024/Lambda)

#### 4.3. API Management - AWS API Gateway

El ultimo paso de esta etapa es crear el API Gateway que desencadena el evento que invoca la función Lambda simplemente pasando los datos de prueba a través de un evento. Amazon API Gateway es un servicio completamente administrado que facilita a los desarrolladores la creación, la publicación, el mantenimiento, el monitoreo y la protección de API a cualquier escala. Las

API actúan como la "puerta de entrada" para que las aplicaciones accedan a los datos, la lógica empresarial o la funcionalidad de sus servicios de backend. Con API Gateway, puede crear API RESTful y API WebSocket que permiten aplicaciones de comunicación bidireccional en tiempo real. API Gateway admite cargas de trabajo en contenedores y sin servidor, así como aplicaciones web [AWS, 2023a]. En la Figura 56 se aprecia el funcionamiento de API Gateway.

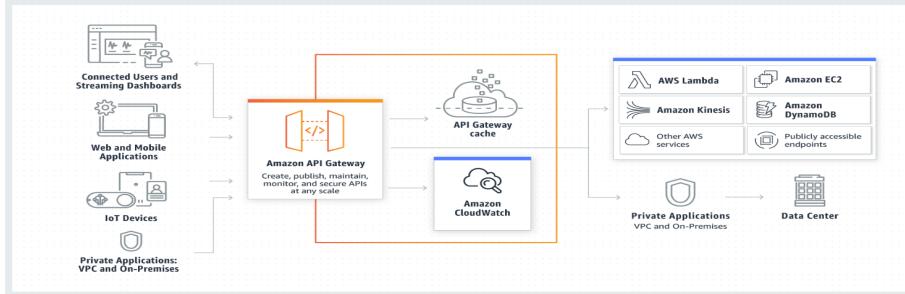


Figura 56: Funcionamiento de API Gateway.

Para crear una “API REST” en el panel de control de AWS se debe buscar el servicio de “API Gateway” como muestra la Figura 57.



Figura 57: Búsqueda Servicio API Gateway.

Tras ingresar al panel de control de “API Gateway” se debe escoger el tipo de API que se va a utilizar, para el desarrollo de este trabajo se escoge la opción de “API REST”, como muestra la Figura 58 y en esta se escoge la opción “Crear”.

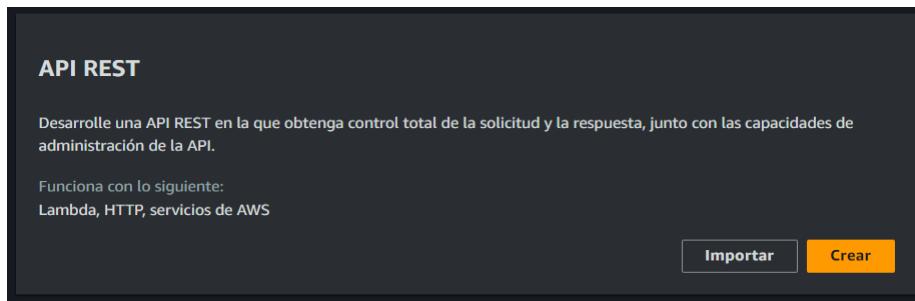


Figura 58: Selección API REST.

Tras elegir la opción de “Crear” se redirigirá al usuario a una nueva pantalla, donde se debe configurar varios parámetros como: Tipo API: se escoge “Nueva API”, Nombre de API, Descripción: es una descripción ligera de la API, es opcional y Tipo de punto de conexión de la API: “Regional”.

Al finalizar con la configuración de los parámetros, ver Figura 59, se presiona el botón de “Crear API” lo cual redirigirá al marco de trabajo de la API donde se puede apreciar que esta se ha creado con éxito.

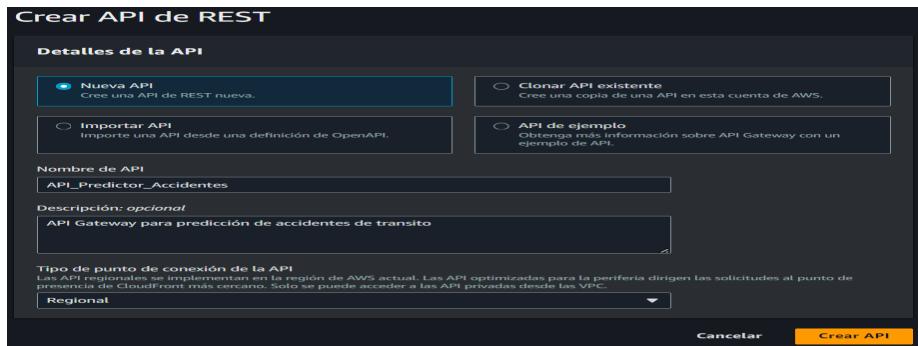


Figura 59: Crear API Gateway.

Una vez creada el API, en el menú “Recurso”, se elige “Crear recurso”, ver Figura 60. Esto abrirá una interfaz donde se configurara los “Detalles del recurso”. En la Figura 61 se muestra la configuración del recurso creado.

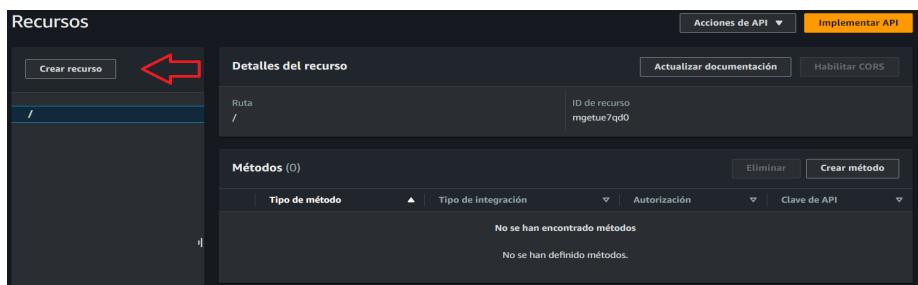


Figura 60: Crear Recurso API.

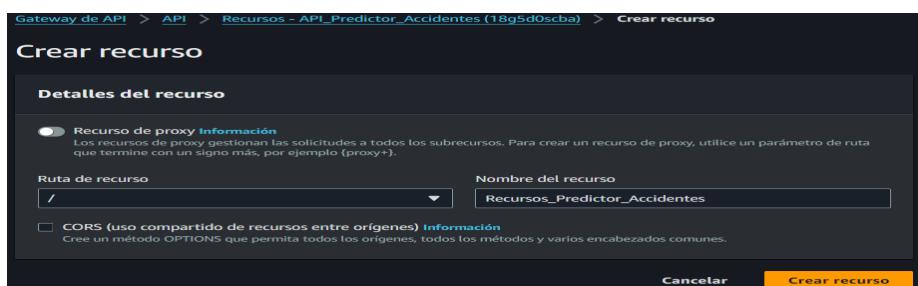


Figura 61: Detalles del Recurso.

Una vez creado el recurso, en el menú “Métodos”, se elije “Crear método” para crear un método POST, ver Figura 62. Esto abrirá una interfaz donde se configurara los “Detalles del método”. En estos detalles se configura el “Tipo de integración”: Función Lambda, la “Región” y el “ARN” de la función Lambda, el ARN se encuentra detallado en la Figura 46, y el tiempo de espera predeterminado que se establece es de 29 segundos por defecto. Ver Figura 63.



Figura 62: Crear Método POST.

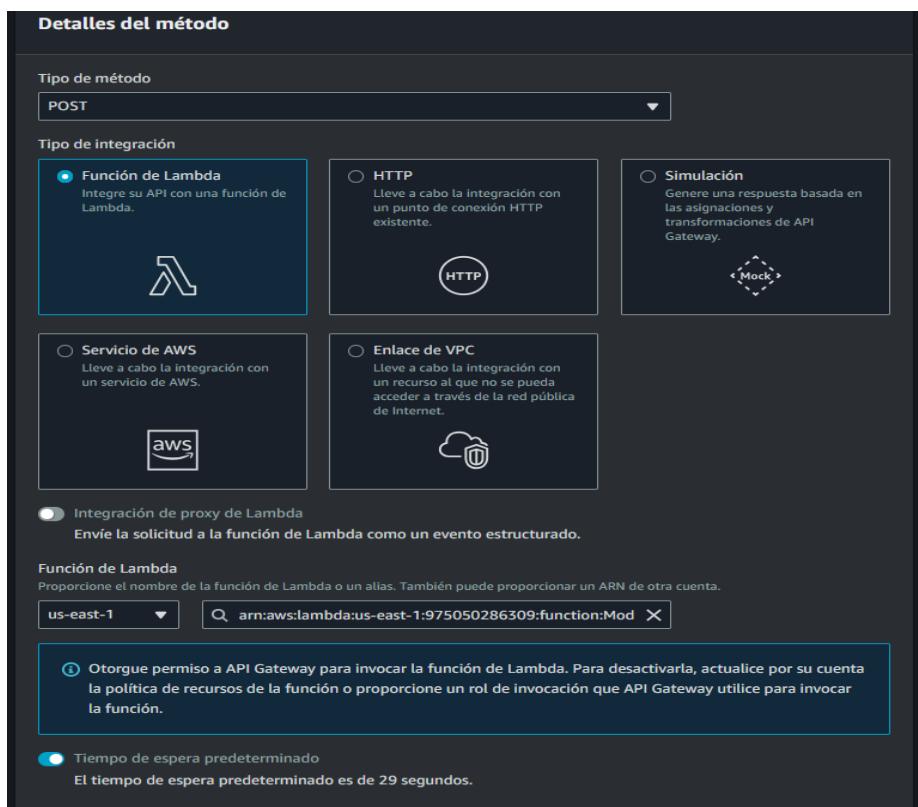


Figura 63: Detalles Método POST.

Al finalizar solo basta con desplegar la API, para ello se presiona el botón “Implementar API” como muestra la Figura 64. Esto lleva a una interfaz donde se detalla la información final sobre la API como: Etapa, Nombre de la Etapa y Descripción de la implementación. Ver Figura 65. Este paso le proporciona la URL de invocación. Ver Figura 66.

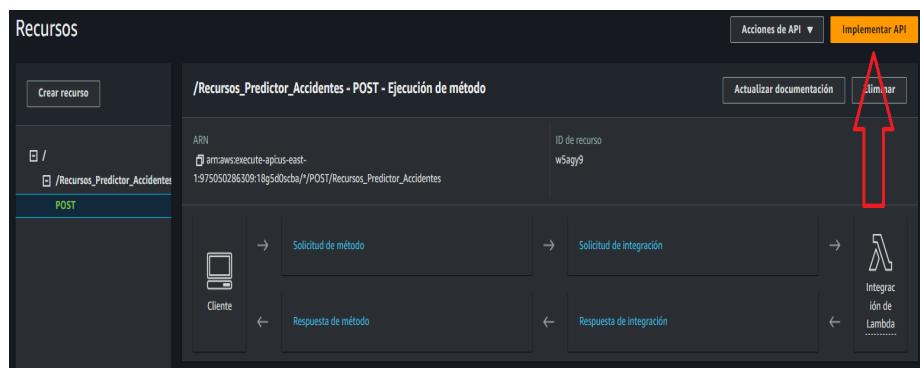


Figura 64: Deploy API Gateway.

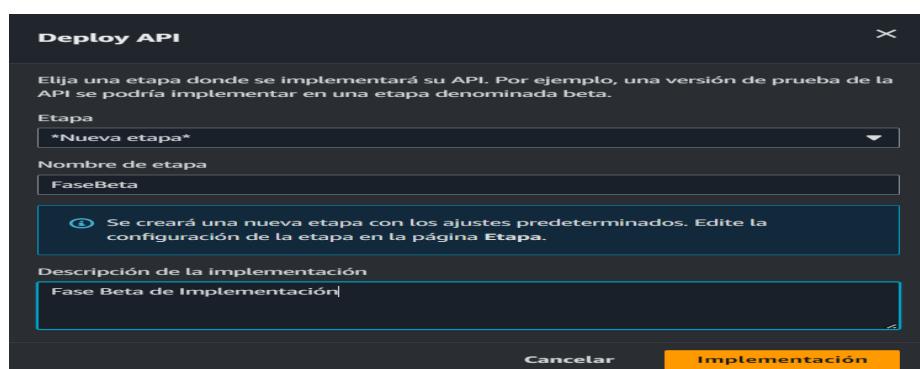


Figura 65: Detalles Deploy API Gateway.

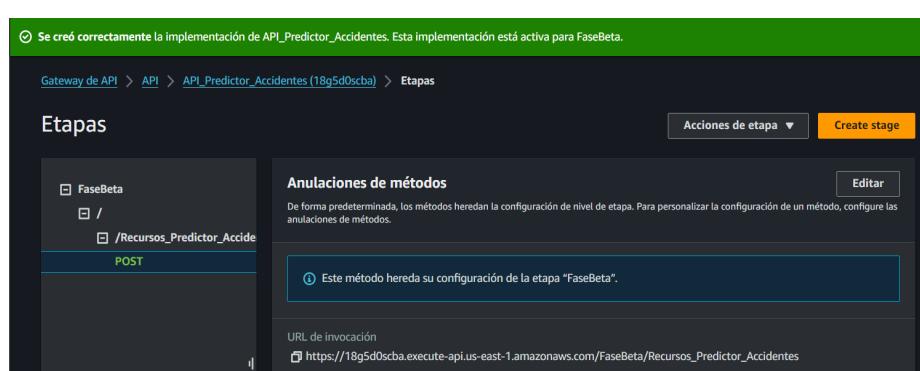


Figura 66: API Implementada Completamente.

#### 4.4. Pruebas de Funcionamiento POST/REST

Tras realizar todas las implementaciones, solo falta realizar las pruebas de funcionamiento. Para realizar las pruebas de funcionamiento se hará uso del entorno JupyterLab. En la Figura 67 se muestra el código implementado en Python, mientras que la Figura 68 se muestran los resultados de las consultas POST/REST realizadas hacia el agente.

```
import pandas as pd
import numpy as np
import requests
import json
import time

def Agente():
    filename_dataset = '20221215_151443_pre.csv'
    df = pd.read_csv(filename_dataset)
    X = df[['steering_angle', 'speed', 'rpm', 'acceleration', 'throttle_position', 'engine_temperature', 'system_voltage', 'heart_rate', 'distance_travelled', 'latitude', 'longitude', 'current_weather', 'accidents_onsite']]
    url = 'https://18g5dscba.execute-api.us-east-1.amazonaws.com/FaseBeta/Recursos_Predictor_Accidentes'
    counter = 0
    while counter < 50:
        random_number = np.random.randint(0, 2633 + 1)
        data = X.loc[[random_number]].values.tolist()
        my_obj = {"Input": data}
        response = requests.post(url, data = json.dumps(my_obj))
        risk_level_values = {1: 'low', 2: 'medium', 3: 'high', 4: 'very high'}
        risk_level = eval(response.text)
        risk_level_category = ""
        for value in risk_level_values:
            if value == risk_level['Output']:
                risk_level_category = risk_level_values[value]
                break
        print("El valor riesgo para: '{}' es '{}'".format(my_obj['Input'], risk_level_category.upper()))
        print(risk_level)
        print(" ")
        counter = counter + 1
        time.sleep(0.25)
Agente()
```

Figura 67: Código Cliente JupyterLab.

```
[1]: %run Cliente.py
El valor riesgo para: '[[[-44.0, 72.0, 4176.0, 0.139095383, 25.88235294, 90.0, 12.9, 65.0, 7.690762379, -0.32757475, -78.38102267, 2.0, 6.0]]' es 'HIGH'.
('Output': 3)

El valor riesgo para: '[[-8.9, 49.0, 1544.0, 0.579512284, 35.29411765, 89.0, 12.8, 68.0, 34.44054331, -0.134915533, -78.35847887, 1.0, 4.0]]' es 'MEDIUM'.
('Output': 2)

El valor riesgo para: '[[[-137.6, 57.0, 4287.0, 0.0, 18.82352941, 87.0, 12.9, 68.0, 24.95850918, -0.215611761, -78.33273998, 2.0, 7.0]]' es 'MEDIUM'.
('Output': 2)

El valor riesgo para: '[[292.8, 61.0, 3514.5, 0.0, 66.2745098, 89.0, 12.8, 66.0, 1.700666546, -0.329386792, -78.41825793, 2.0, 9.0]]' es 'VERY HIGH'.
('Output': 4)

El valor riesgo para: '[[[-1.7, 44.0, 1699.0, 0.0, 15.29411765, 88.0, 12.9, 65.0, 10.82446463, -0.3077733, -78.36410338, 2.0, 5.0]]' es 'LOW'.
('Output': 1)

El valor riesgo para: '[[[62.4, 70.0, 4063.5, 0.568192769, 25.49019608, 89.0, 12.8, 63.0, 32.59779911, -0.150818937, -78.35174359, 1.0, 1.0]]' es 'MEDIUM'.
('Output': 2)

El valor riesgo para: '[[[511.3, 2.0, 1139.0, 0.0, 15.29411765, 90.0, 12.7, 76.0, 36.33122785, -0.125206523, -78.36174077, 1.0, 2.0]]' es 'MEDIUM'.
('Output': 2)

El valor riesgo para: '[[[106.4, 83.0, 4786.5, 0.0, 19.21568627, 88.0, 12.8, 70.0, 24.05208214, -0.222700294, -78.32798718, 2.0, 1.0]]' es 'HIGH'.
('Output': 3)

El valor riesgo para: '[[[-130.4, 70.0, 4062.0, 0.207257625, 18.43137255, 88.0, 12.9, 66.0, 4.482338368, -0.323055186, -78.40615103, 2.0, 5.0]]' es 'HIGH'.
('Output': 3)

El valor riesgo para: '[[[-28.2, 41.0, 2918.0, 1.24656422, 33.7254902, 89.0, 12.8, 69.0, 22.01368512, -0.239808303, -78.3306191, 2.0, 17.0]]' es 'MEDIUM'.
('Output': 2)

El valor riesgo para: '[[[65.0, 0.0, 661.0, 0.0, 13.7254902, 90.0, 12.7, 74.0, 36.33168684, -0.125208409, -78.36173968, 1.0, 2.0]]' es 'MEDIUM'.
('Output': 2)
```

Figura 68: Resultados Consulta POST/REST.

En el siguiente enlace se encuentra el código de implementación para las pruebas de funcionamiento:

- [https://github.com/laboratorioAI/PIS\\_20\\_02\\_AGENTES\\_IA/tree/main/Actualizacion\\_2024/Cliente](https://github.com/laboratorioAI/PIS_20_02_AGENTES_IA/tree/main/Actualizacion_2024/Cliente)

## 5. Implementación APP Móvil - Android Studio

La implementación de una aplicación móvil en Android Studio implica un proceso completo de desarrollo que abarca desde la concepción de la idea hasta la entrega final del producto. Android Studio, siendo el entorno de desarrollo oficial respaldado por Google, juega un papel central en este proceso, proporcionando herramientas poderosas y funcionalidades que facilitan la creación de aplicaciones Android robustas y efectivas [Studio, 2024].

El primer paso en la implementación de una aplicación es la configuración del entorno de desarrollo. Esto implica descargar e instalar Android Studio, así como configurar el entorno con los componentes necesarios, como los paquetes de desarrollo de software (SDK) y las herramientas específicas de Android. Este paso es crucial para garantizar que el desarrollador tenga acceso a las bibliotecas y recursos necesarios para construir y probar la aplicación de manera efectiva.

### 5.1. Aplicaciones Web y Móviles - AWS Amplify

AWS Amplify Hosting es un servicio de alojamiento y CI/CD completamente administrado para aplicaciones estáticas, rápidas, seguras, fiables, renderizadas del lado del servidor y que escalan con su empresa. Es compatible con marcos web modernos como React, Angular, Vue, Next.js, Gatsby, Hugo, Jekyll, entre otros [AWS, 023s]. En la Figura 69 se aprecia el funcionamiento de AWS Amplify.

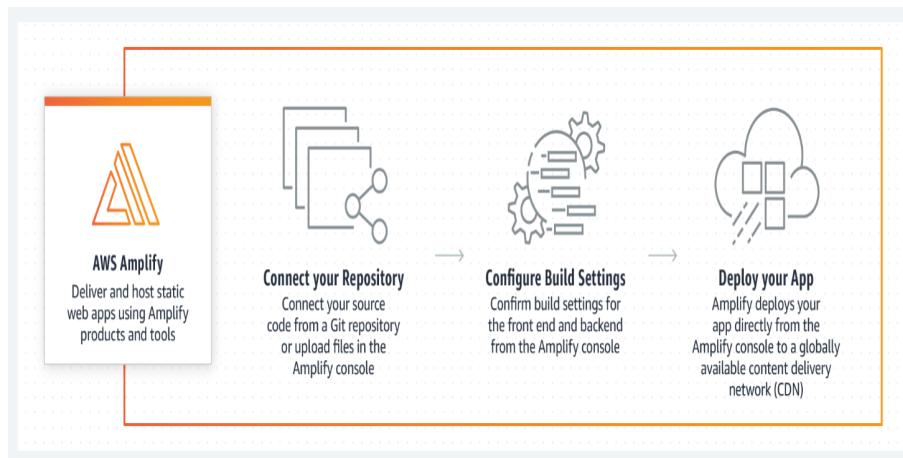


Figura 69: Funcionamiento de AWS Amplify.

El primer paso es instalar Node.js en el computador de trabajo desde la siguiente URL: <https://nodejs.org/en/download/>, de existir problemas se puede encontrar documentación en la siguiente pagina <https://docs.npmjs.com/downloading-and-installing-node-js-and-npm>. Ver Figura 70.

El siguiente paso es el instalar la consola de “Amplify” mediante el comando de la Figura 71, para ello se debe abrir una terminal en modo administrador y ejecutar el comando.



Figura 70: Repositorio Nodes.

De preferencia y por seguridad se debe abrir dentro del entorno de desarrollo de “Android Studio” ejecutándolo como “Administrador”. Ver Figura 71.



Figura 71: Comando de Instalación de Amplify.

Continuando, se debe abrir Android Studio en modo Administrador. Posteriormente, en el proyecto en el cual se está trabajando se debe abrir la consola de comando. Ver Figura 72. Para ver como crear un nuevo proyecto ver el apartado 5.2.

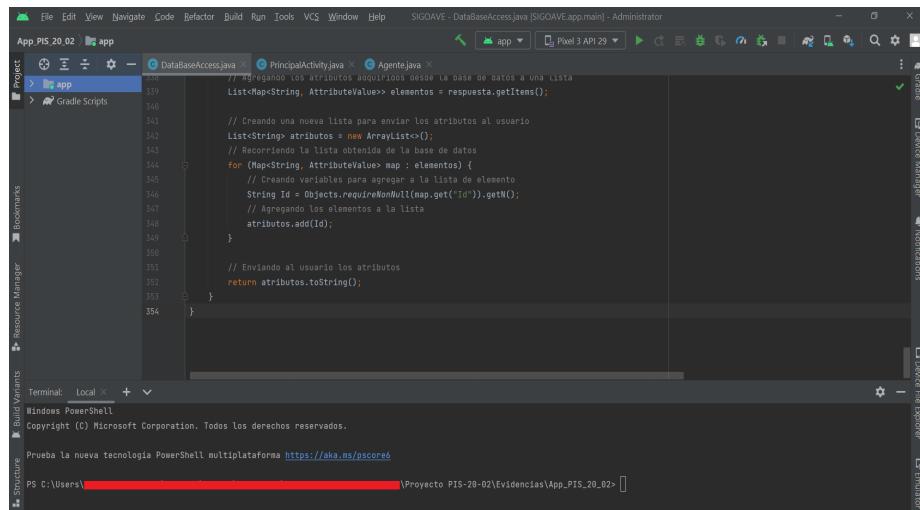


Figura 72: Consola de Comandos de Android.

En la consola de comandos se debe ejecutar el comando: “amplify configure”, esto abrirá una ventana en el navegador de su preferencia donde debe iniciar sesión en su cuenta de AWS como usuario raíz o tener los permisos suficientes para realizar cambios en la estructura de AWS, ver Figura 73.

```
Terminal: Local × + ▾
PS C:\Users\██████████\MyApplication> amplify configure
Follow these steps to set up access to your AWS account:

Sign in to your AWS administrator account:
https://console.aws.amazon.com/
Press Enter to continue
```

Figura 73: Sesión de Amplify.

Iniciada sesión se debe regresar al entorno de Android Studio, aquí se debe escoger la región de AWS en la cual se va a trabajar, ver Figura 74. Escogida la región, se abrirá una nueva ventana en el navegador en la cual se debe crear un nuevo usuario para la aplicación. Para la creación nuevo usuario ver el apartado 4.1. Dato: se puede usar un usuario anteriormente creado.

```
Sign in to your AWS administrator account:
https://console.aws.amazon.com/
Press Enter to continue

Specify the AWS Region
? region: (Use arrow keys)
> us-east-1
  us-east-2
  us-west-1
  us-west-2
  eu-north-1
  eu-south-1
  eu-west-1
(Move up and down to reveal more choices)
```

Figura 74: Región de Amplify.

Cabe aclarar, en la consola de IAM cuando se debe escoger las políticas que se van a adjuntar al usuario que se esta creando en la opción “Adjuntar políticas directamente”, se debe escoger: “AdministratorAccess-Amplify”. Ver Figura 75.

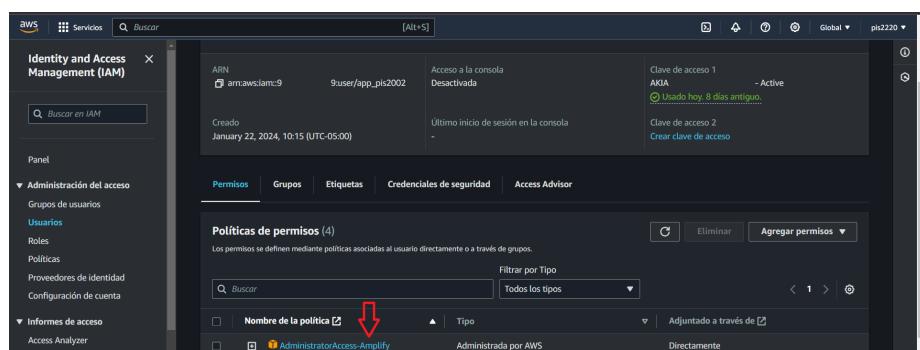


Figura 75: Detalle de Usuario de IAM para Amplify.

Siguiendo los pasos del apartado 4.1, llegamos al punto donde obtenemos las “Claves de acceso”: “Clave de acceso” y “Clave de acceso secreta” como muestra la Figura 15. Con las claves creadas, retornamos al entorno de Android e ingresamos las claves en la consola de comandos como muestra la Figura 76.

```
to complete the user creation in the AWS console  
https://console.aws.amazon.com/iamv2/home#/users/create  
Press Enter to continue  
  
Enter the access key of the newly created user:  
? accessKeyId: ****  
? secretAccessKey: ****  
This would update/create the AWS Profile in your local machine
```

Figura 76: Amplify Access y Secret Key.

Posteriormente, ingresamos el nombre del perfil mediante consola como muestra la Figura 77. En esta podemos ver como se ha creado exitosamente un usuario de “Android” en AWS Amplify.

```
Enter the access key of the newly created user:  
? accessKeyId: ****  
? secretAccessKey: ****  
This would update/create the AWS Profile in your local machine  
? Profile Name: pis_2002  
  
Successfully set up the new user.
```

Figura 77: Amplify Nombre de Perfil.

## 5.2. Creación Nuevo Proyecto - Android Studio

Una vez configurado el entorno, se inicia la creación de un nuevo proyecto en Android Studio. Durante este proceso, se elige el tipo de actividad principal que servirá como punto de partida para la aplicación, como una actividad en blanco o una actividad de navegación. Esta elección inicial establece la estructura básica del proyecto y define cómo interactuará la aplicación con los usuarios.

El diseño de la interfaz de usuario es una parte esencial del desarrollo de aplicaciones móviles. Android Studio ofrece un Editor de diseño que permite a los desarrolladores visualizar y diseñar la apariencia de la aplicación de manera intuitiva. La disposición de los elementos, la elección de colores y el estilo de la interfaz de usuario son consideraciones clave durante esta fase.

La aplicación móvil del agente de respuesta incluye lo siguiente:

- Una pantalla o “activity” para gestionar el inicio de sesión por medio de credenciales (nombre de usuario y contraseña).
- Pantallas para presentar los siguientes parámetros claves:
  - Vehículo: velocidad, revoluciones por minuto, nivel de gasolina, temperatura del motor y ángulo de inclinación del volante.
  - Condiciones meteorológicas: lugar o zona, temperatura, precipitación, humedad, y velocidad del viento.
  - Flujo de tráfico: ubicación, velocidad promedio, número de vehículos, ocupación promedio y dirección del flujo.
  - Conductor: edad, genero, uso de lentes, estado de licencia y puntos disponibles, y condiciones médicas.
- Una pantalla que incluye un mapa para ubicar los vehículos que se encuentran en la zona.
- Cada una de las pantallas presenta como encabezado información relevante al usuario.

La programación constituye el núcleo del desarrollo de la aplicación, donde se implementa la lógica y la funcionalidad. La programación implica gestionar eventos, realizar operaciones de entrada/salida, conectarse a servicios web, entre otras tareas que definen el comportamiento y la interactividad de la aplicación.

Los pasos a seguir en cada una de las etapas de desarrollo del “Agente de Respuesta – SIGOAVE” están documentadas en el documento “Desarrollo de Aplicación Móvil en Android Studio como Agente de respuesta del Proyecto PIS2002” ubicado en el siguiente enlace:

- [https://github.com/laboratorioAI/PIS\\_20\\_02\\_AGENTES\\_IA/tree/main/Actualizacion\\_2024/APP%20PIS2002%20Documentacion](https://github.com/laboratorioAI/PIS_20_02_AGENTES_IA/tree/main/Actualizacion_2024/APP%20PIS2002%20Documentacion)

### 5.3. Implementación de AWS Amplify en el Proyecto

Finalizada la creación base del proyecto, se necesita al menos una “Activity” en el proyecto, se continua con la configuración de Amplify en el proyecto que se esta desarrollando.

El primer paso para la incorporación de Amplify es la instalación de la bibliotecas para Amplify. Los servicios de AWS Amplify para Android se distribuye como paquetes Apache Maven [AWS, 023s]. En esta sección, se agregará los paquetes y otras directivas necesarias para la configuración de compilación. En Gradle Scripts, se debe abrir “build.gradle (Module :app)” y agregar las siguientes lineas: “coreLibraryDesugaringEnabled”, “sourceCompatibility”, y “targetCompatibility” para permitir que su aplicación utilice funciones de Java 8 como expresiones Lambda. Agregar las bibliotecas Amplify Core y Desugaring al bloque “dependencies”. Ver Figura 78.

```

    android {
        compileOptions {
            // Support for Java 8 features
            coreLibraryDesugaringEnabled true
            sourceCompatibility JavaVersion.VERSION_1_8
            targetCompatibility JavaVersion.VERSION_1_8
        }
    }

    dependencies {
        // Amplify core dependency
        implementation 'com.amplifyframework:core:2.14.6'

        // Support for Java 8 features
        coreLibraryDesugaring 'com.android.tools:desugar_jdk_libs:1.1.5'
    }
}

```

Figura 78: Librerías Amplify en Gradle.

Ejecutar la sincronización de Gradle. Android Studio requiere que sincronice el proyecto con la nueva configuración. Para hacer esto, hacer “Clic” en “Sync Now” en la barra de notificaciones encima del editor de archivos. Ver Figura 79.

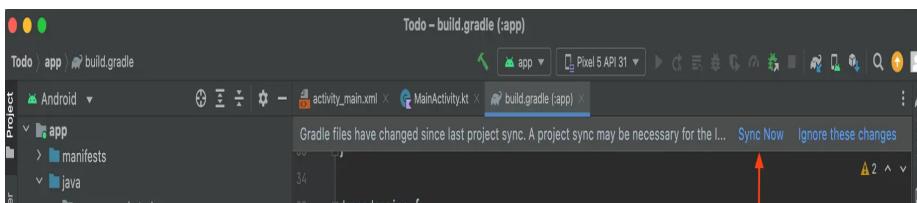


Figura 79: Sync Gradle.

Cuando termine, se verá “CONFIGURACIÓN EXITOSA” en el resultado de la pestaña “Construir” en la parte inferior de la pantalla. Ver Figura 80

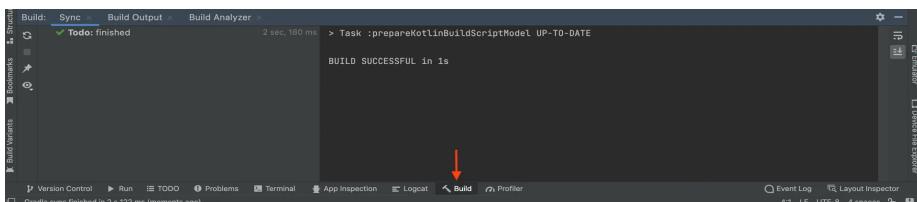


Figura 80: Sync Éxito.

El siguiente paso es el aprovisionamiento de “Backend con Amplify CLT”, para comenzar a aprovisionar recursos en el backend, se debe abrir un terminal de trabajo en Android Studio y ejecutar: “amplify init”. Ver Figura 81.

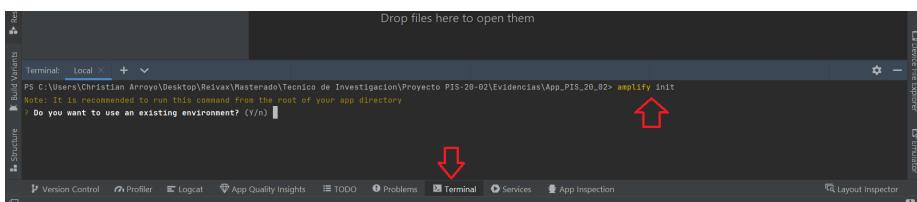


Figura 81: Amplify Init.

Tras iniciar Amplify se debe ingresar lo siguiente cuando se le solicite, ver Figura 82. Este proceso tardara unos minutos mientras Android crea el proyecto en la cuenta de AWS.

```
Enter a name for the project
`MyAmplifyApp`
Initialize the project with the above configuration?
`No`
Enter a name for the environment
`dev`
Choose your default editor:
`Android Studio`
Choose the type of app that you're building
`android`
Where is your Res directory:
`app/src/main/res`
Select the authentication method you want to use:
`AWS profile`
Please choose the profile you want to use
`default`
```

Figura 82: Amplify Init Configuración.

Al ejecutar exitosamente “amplify init”, se verá un archivo de configuración creado en el directorio del proyecto: “./app/src/main/res/raw/” llamado “amplifyconfiguration.json”. Este archivo se incluirá en su aplicación para que las bibliotecas de Amplify sepan cómo acceder a sus recursos backend aprovisionados en tiempo de ejecución, ver Figura 83.

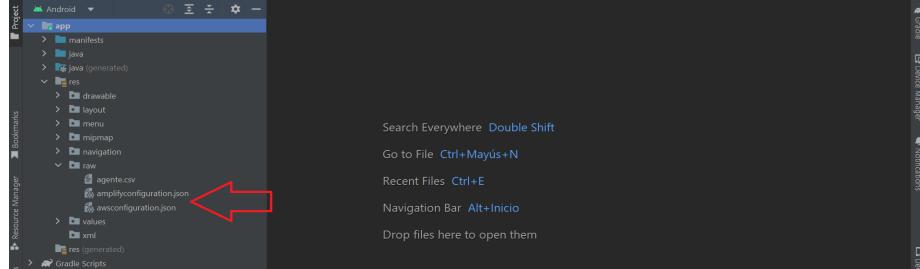


Figura 83: Archivos de Configuración de Amplify.

También, se puede observar que dentro del entorno de Amplify y S3 en AWS se ha creado el proyecto y un bucket respectivamente, ver Figura 84 y 85.



Figura 84: Amplify AWS Proyecto.

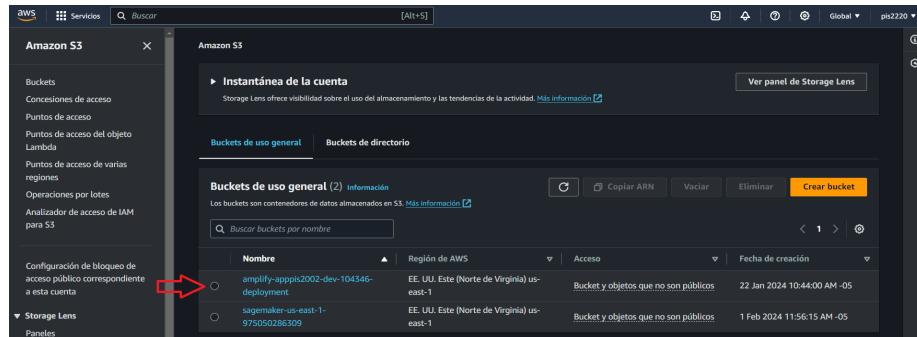


Figura 85: S3 AWS Proyecto.

El siguiente paso es inicializar Amplify dentro de la aplicación. Para ello en la ubicación deseada dentro del proyecto se debe crear una clase a la cual llamaremos “PrincipalAmplify”, esta clase se tiene que extender desde “Application” y sobre-escribir “onCreate()” para inicializar Amplify en la aplicación. Ver Figura 86.

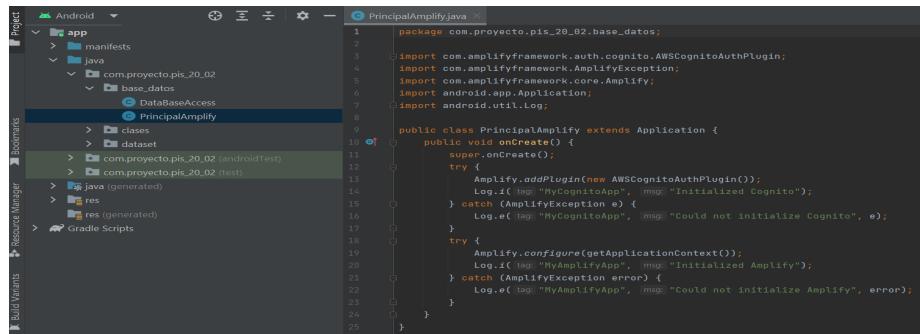


Figura 86: Principal Amplify.

A continuación, configurar la aplicación para usar la nueva clase “PrincipalAmplify” personalizada. Abrir el “AndroidManifest.xml” y agregar un atributo “android:name” con el valor de su nuevo nombre de clase, ver Figura 87.

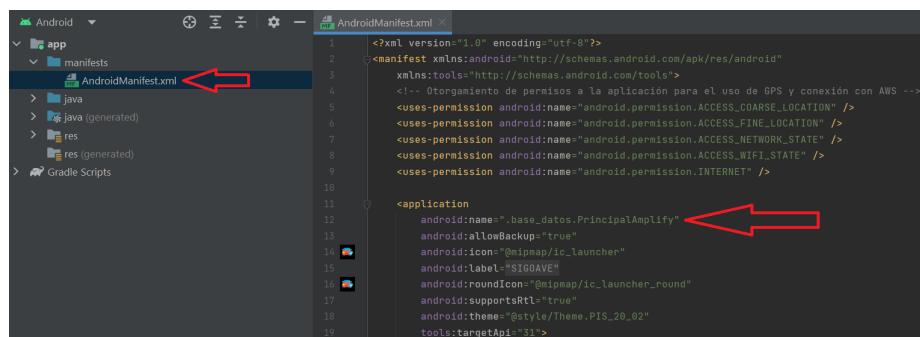


Figura 87: Configuración Manifest.

## 5.4. Customer IAM - AWS Cognito

Con Amazon Cognito, puede agregar funciones de registro e inicio de sesión para los usuarios y controlar el acceso a sus aplicaciones web y móviles. Amazon Cognito brinda un almacenamiento de identidades que es escalable a millones de usuarios, respalda la federación de identidades social y empresarial, y ofrece funciones de seguridad avanzada para proteger a sus clientes y a su empresa. Creado en base a las normas de identidad abierta, Amazon Cognito es compatible con múltiples regulaciones de conformidad y se integra con los recursos de desarrollo de frontend y backend [AWS, 2023b]. En la Figura 88 se aprecia el funcionamiento de Cognito.

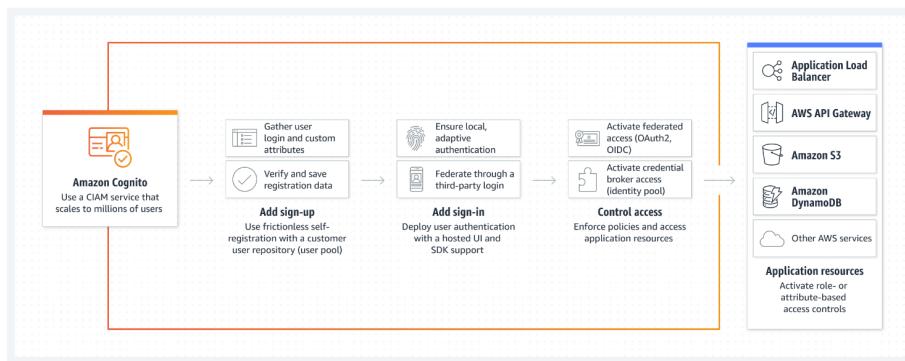


Figura 88: Funcionamiento de Cognito.

Instalado AWS Amplify en el proyecto de Android Studio se procede con la instalación y configuración de AWS Cognito, esto permite configurar la autenticación de Amplify. La categoría “Amplify Auth” proporciona una interfaz para autenticar a un usuario. Detrás de escena, proporciona la autorización necesaria a las otras categorías de Amplify. Viene con soporte integrado predeterminado para el grupo de usuarios y el grupo de identidades de Amazon Cognito . La CLI de Amplify puede ayudar a crear y configurar la categoría de autenticación con un proveedor de autenticación.

Primero, se debe agregar la siguiente dependencia a la aplicación, al igual que con Amplify esta se la agrega en el “build.gradle (Module :app)”, ver Figura 89. Luego los cambios realizados en este archivo se debe sincronizar al finalizar.

```
1 dependencies {  
2     implementation 'com.amplifyframework:aws-auth-cognito:2.14.6'  
3 }
```

Figura 89: Librerías Cognito en Gradle.

Agregado los recurso y para comenzar a aprovisionar recursos de autenticación en el backend, dentro de Android Studio se debe abrir el terminal y ejecutar el siguiente comando: “amplify add auth”. Ver Figura 90.

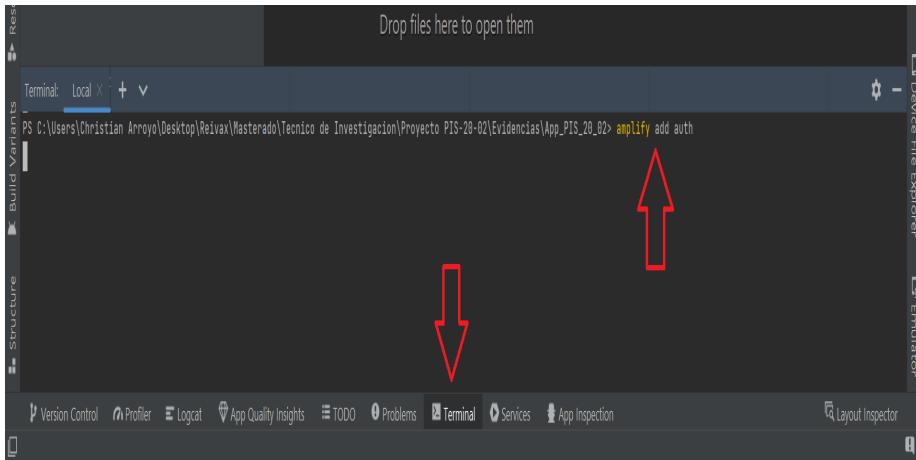


Figura 90: Amplify Add Auth.

Tras ejecutar el comando se debe ingresar lo siguiente cuando se le solicite, ver Figura 91. Este proceso tardara unos minutos mientras Android crea el proyecto en la cuenta de AWS.

```
? Do you want to use the default authentication and security configuration?
`Default configuration`
? How do you want users to be able to sign in?
`Username`
? Do you want to configure advanced settings?
`No, I am done.`
```

Figura 91: Amplify Add Auth Configuración.

Para enviar los cambios realizados a la nube, se debe ejecutar el siguiente comando comando: “amplify push”. Ver Figura 92. Una vez finalizado, “amplifyconfiguration.json” debe actualizarse para hacer referencia a los recursos de autenticación de backend aprovisionados. Se debe tener en cuenta que estos archivos ya deberían ser parte del su proyecto como se muestra en la Figura 83.

```
UPDATE_COMPLETE_CLEANUP_IN_PROGRESS amplify-tutorialandroidampli-dev-200927 AWS::CloudFormation::Stack Mon Feb 21 2022 20:24:33 GMT+0100 (GMT+01:00)
UPDATE_COMPLETE          amplify-tutorialandroidampli-dev-200927 AWS::CloudFormation::Stack Mon Feb 21 2022 20:24:34 GMT+0100 (GMT+01:00)
✓ All resources are updated in the cloud
```

Figura 92: Amplify Push.

Verificar que en la clase “PrincipalAmplify” se encuentre agregado las configuraciones para Cognito Ver Figura 86. Finalmente, comprobar dentro del entorno de AWS en el panel de control de Cognito que se haya creado el proyecto, ver Figura 93.

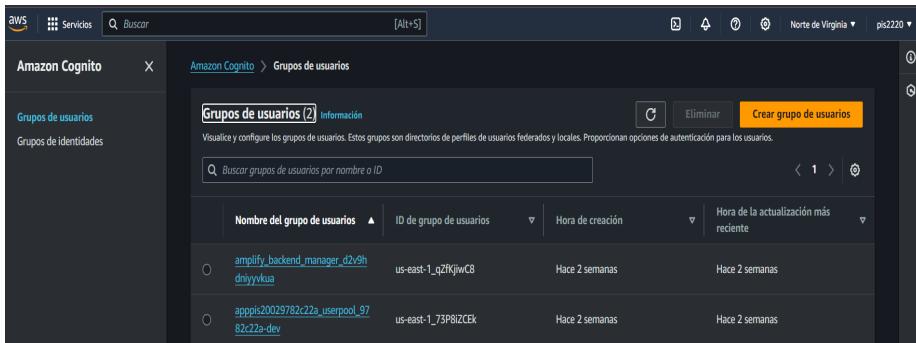


Figura 93: Cognito AWS Proyecto.

### 5.5. Prueba de Funcionamiento Amplify y Cognito

Las pruebas son cruciales para garantizar que la aplicación funcione correctamente en diversas situaciones y dispositivos. Android Studio proporciona un emulador integrado que permite probar la aplicación en diferentes versiones de Android y resoluciones de pantalla. Además, los desarrolladores pueden probar la aplicación en dispositivos físicos para obtener una experiencia más realista.

En esta etapa se probará el funcionamiento de la configuración de Amplify y Cognito, para ello solo se necesita correr la aplicación dentro de Android Studio y en la sección de “Logcat” verificar si se estableció conexión con AWS Amplify. Ver Figura 94.

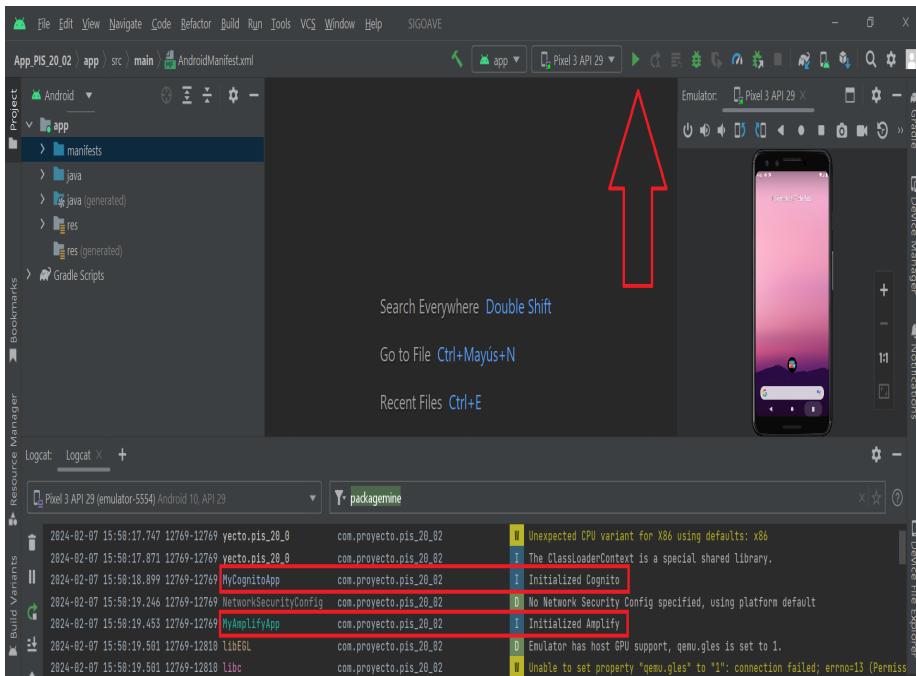


Figura 94: Comprobación funcionamiento de Amplify.

## 5.6. Servicio de Base de Datos Gestionada NoSQL - AWS DynamoDB

Amazon DynamoDB es una base de datos NoSQL de clave-valor sin servidor y completamente administrada que está diseñada para ejecutar aplicaciones de alto rendimiento a cualquier escala. DynamoDB ofrece seguridad integrada, copias de seguridad continuas, replicación automatizada en varias regiones, almacenamiento de caché en memoria y herramientas de importación y exportación de datos [AWS, 2023c]. En la Figura 95 se aprecia el funcionamiento de DynamoDB.

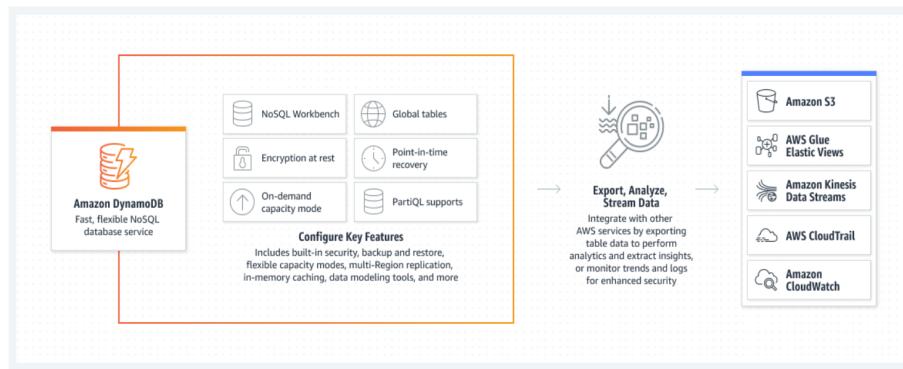


Figura 95: Funcionamiento de DynamoDB.

En esta etapa del proyecto corresponde a la creación de la base de datos de la APP de desarrollada en Android Studio, para ello se hará uso de AWS DynamoDB, tres tablas serán creadas para la aplicación: DataSet (contiene el conjunto de datos del sistema de predicción de accidentes), DataSet\_Alertas (contiene las respuestas o predicciones) y UserData (contiene la información personal de los usuarios).

Para comenzar con el proceso de creación de las tablas, el primer paso es buscar el servicio de AWS DynamoDB en el panel de control de AWS como muestra la Figura 96.

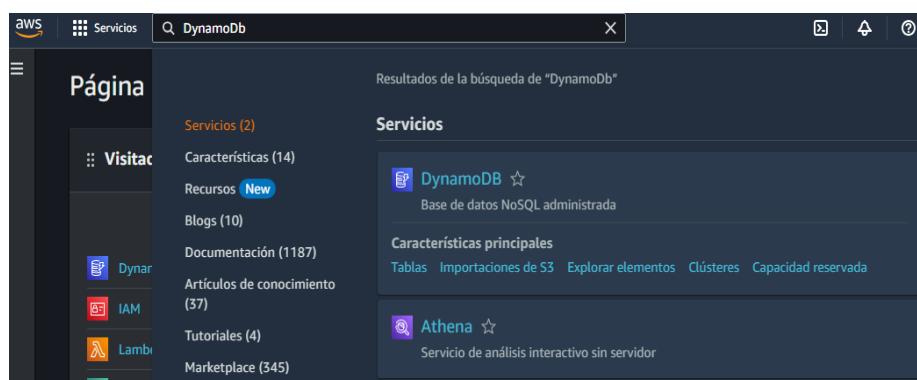


Figura 96: Búsqueda Servicio DynamoDB.

Localizado el servicio, se ingresa al mismo, y dentro del panel de control de DynamoDB se busca la opción “Tablas” y se da “Clic” sobre esta opción como muestra la Figura 97.

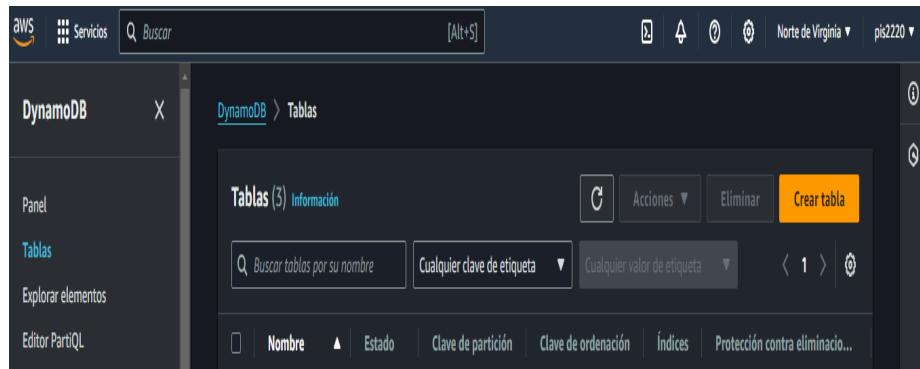


Figura 97: Tablas de DynamoDB.

Al ingresar a “Tablas” se puede observar la opción de “Crear tabla”, al ingresar en esta opción se redirigirá al usuario a una nueva interfaz donde se debe llenar una serie de parámetros que permiten configurar el funcionamiento de la tabla que se está por crear.

Los parámetros que se configuran de la tabla ha crear son:

- Nombre de la tabla: Se utilizará para identificar su tabla (Entre 3 y 255 caracteres. Solo se pueden usar letras, números, guiones bajos (\_), guiones (-) y puntos (.)).
- Clave de partición: La clave de partición forma parte de la clave principal de la tabla. Se trata de un valor hash que se utiliza para recuperar elementos de la tabla, así como para asignar datos entre hosts por cuestiones de escalabilidad y disponibilidad.
- Clave de ordenación - opcional: Puede utilizar una clave de ordenación como segunda parte de la clave principal de una tabla. La clave de ordenación le permite ordenar o buscar entre todos los elementos que comparten la misma clave de partición.
- Configuración de la tabla: Escoger la opción “Personalizar configuración”.
- Clase de tabla: La clase de tabla permite optimizar el costo de la tabla en función de los requisitos de la carga de trabajo y los patrones de acceso a los datos. Escoger la opción “Estándar de DynamoDB”.
- Configuración de capacidad de lectura/escritura: Escoger la opción “Bajo demanda”.
- Protección contra eliminaciones: La protección contra eliminaciones evita que la tabla se elimine involuntariamente. Marcar la opción “Activar la protección contra eliminaciones”.
- El resto de parámetro se deben dejarlos en la opción predefinida.

La Figura 98 muestra un ejemplo de las configuraciones realizadas para crear una tabla, esta configuraciones se deben realizar en todas las tablas que se van a usar en el proyecto.

**Crear tabla**

**Detalles de la tabla** [información](#)  
DynamoDB es una base de datos sin esquemas que solo requiere un nombre de tabla y una clave principal al crear la tabla.

**Nombre de la tabla**  
Se utilizará para identificar su tabla.  
**Tabla\_Ejemplo**  
Entre 3 y 255 caracteres. Solo se pueden usar letras, números, guiones bajos (-), guiones (-) y puntos (.)

**Clave de partición**  
La clave de partición forma parte de la clave principal de la tabla. Se trata de un valor hash que se utiliza para recuperar elementos de la tabla, así como para asignar datos entre hosts por cuestiones de escalabilidad y disponibilidad.  
**Id** **Número**

**Clave de ordenación - opcional**  
Puede utilizar una clave de ordenación como segunda parte de la clave principal de una tabla. La clave de ordenación le permite ordenar o buscar entre todos los elementos que comparten la misma clave de partición.  
**Nombre** **Cadena**

**Configuración de la tabla**

- Configuración predeterminada**  
La forma más rápida de crear la tabla. Puede modificar estos ajustes ahora o después de que se haya creado la tabla.
- Personalizar configuración**  
Utilice estas características avanzadas para que DynamoDB funcione mejor de acuerdo a sus necesidades.

**Clase de tabla**  
Seleccione la clase de tabla para optimizar el costo de la tabla en función de los requisitos de la carga de trabajo y los patrones de acceso a los datos.

**Elegir la clase de tabla**

- Estándar de DynamoDB**  
La clase de tabla de uso general predeterminada. Se recomienda para las tablas que almacenan datos a los que se accede con frecuencia, donde el almacenamiento es el principal costo de la tabla.
- DynamoDB Standard-IA**  
Se recomienda para las tablas que almacenan datos a los que se accede con menor frecuencia, donde el almacenamiento es el principal costo de la tabla.

**▶ Calculadora de capacidad**

**Configuración de capacidad de lectura/escritura** [información](#)

**Modo de capacidad**

- Aprovisionado**  
Administre y optimice los costos asignando la capacidad de lectura/escritura por adelantado.
- Bajo demanda**  
Simplifique la facturación pagando por las lecturas y escrituras reales que realiza su aplicación.

**Índices secundarios** [información](#)

Nombre	Tipo	Clave de partición	Clave de ordenación	Atributos proyectados
No hay índices Use los índices secundarios para hacer consultas sobre atributos que no forman parte de la clave principal de una tabla.				
<a href="#">Crear índice global</a>				

**Cifrado en reposo** [información](#)  
Todos los datos de usuario almacenados en Amazon DynamoDB están totalmente cifrados en reposo. De forma predeterminada, Amazon DynamoDB administra la clave de cifrado y no se le cobrará ninguna cuota por utilizarla.

**Administración de claves de cifrado**

- Propiedad de Amazon DynamoDB** [Más información](#)  
DynamoDB posee y administra la clave de AWS KMS. No se cobrará ningún cargo adicional por el uso de esta clave.
- Clave administrada por AWS** [Más información](#)  
La clave se almacena en la cuenta y es administrada por AWS Key Management Service (AWS KMS). Se aplican cargos de AWS KMS.
- Es suya, se almacena en su cuenta y es su responsabilidad administrarla.** [Más información](#)  
La clave es suya, se almacena en su cuenta y es su responsabilidad administrarla. AWS KMS cobra cargos.

**Protección contra eliminaciones** [información](#)

**Activar la protección contra eliminaciones**

**Etiquetas**  
Las etiquetas son pares de claves y valores opcionales que puede asignar a los recursos de AWS. Puede utilizar etiquetas para controlar el acceso a los recursos o realizar un seguimiento de los gastos en AWS.

No hay etiquetas asociadas al recurso.

[Agregar nueva etiqueta](#)  
Puede agregar 50 etiquetas más.

[Cancelar](#) [Crear tabla](#)

Figura 98: Configuración de las Tabla en DynamoDB.

Tras realizar la configuración de la Tabla se presiona sobre el botón “Crear Tabla” lo cual redirigirá al panel de control principal de la creación de tablas y se podrá observar que la tabla ya se encuentra, ver Figura 99.

The screenshot shows the AWS DynamoDB console with the title bar "aws | Servicios | Buscar [Alt+S]". Below it is a green banner stating "La tabla Tabla\_Ejemplo se ha creado correctamente." The main area is titled "DynamoDB > Tablas" and shows a table with four rows:

	Nombre	Estado	Clave de partición	Clave de ordenación	Índices	Protección contra eliminación...	Modo de capacidad...
<input type="checkbox"/>	DataSet	Activo	Id (N)	time (\$)	0	Activada	Bajo demanda
<input type="checkbox"/>	DataSet_Alertas	Activo	Alerta (\$)	Id (N)	0	Activada	Bajo demanda
<input type="checkbox"/>	Tabla_Ejemplo	Activo	Id (N)	Nombre (\$)	0	Activada	Bajo demanda
<input type="checkbox"/>	UserData	Activo	Correo (\$)	UserName (\$)	0	Activada	Bajo demanda

Figura 99: Tablas DynamoDB Creadas.

Si se ingresa a la tabla dando “Clic” sobre el nombre de la tabla se cambiara de a la interfaz donde se muestran los detalles de la tabla que muestra la “Información general” de la tabla. En este apartado interesa la opción “Explore los elementos de la tabla” lo cual permitirá crear elementos dentro de la tabla, ver Figura 100.

The screenshot shows the "Explorar elementos" interface for the "Tabla\_Ejemplo" table. The left sidebar lists tables: DataSet, DataSet\_Alertas, Tabla\_Ejemplo (selected), and UserData. The main area has sections for "Escanear o consultar elementos" (with "Examen" selected) and "Filtros". At the bottom, a green banner says "Completado. Unidades de capacidad de lectura consumidas: 2". The footer has buttons for "Acciones" and "Crear elemento".

Figura 100: Crear Elementos en Tabla de DynamoDB.

En la interfaz de “Crear elemento” se deben ingresar los atributos de la tabla en el formato que esta mismo solicita, ver Figura 101. Se puede ingresar los datos ya sea en forma de “Formulario” o en forma de “JSON”.

DynamoDB > Explorar elementos: Tabla\_Ejemplo > Crear elemento

### Crear elemento

Puede agregar, eliminar o editar los atributos de un elemento. Es posible anidar atributos dentro de otros atributos hasta 32 niveles de profundidad. [Más información](#)

Atributos	Valor	Tipo
Id - Clave de partición	1	Número
Nombre - Clave de ordenación	Christian	Cadena

[Agregar nuevo atributo](#)

[Cancelar](#) [Crear elemento](#)

Figura 101: Nuevo Elemento Tabla de DynamoDB.

Tras presionar en el botón de “Crear elemento” se regresara al la interfaz principal de “Tablas” donde se podrá ver que la nueva tabla se ha creado con éxito, ver Figura 102.

DynamoDB > Explorar elementos > Tabla\_Ejemplo

Tables (4)

Escanear o consultar elementos

Filtros

Elementos devueltos (1)

	Id (Número)	Nombre (Cadena)
1	Christian	

Figura 102: Elemento Creado DynamoDB.

Este proceso se repetirá para cada tabla que se cree y para cada elemento dentro de las tablas si se desea crearlos de forma manual.

Existe un caso particular para el cual se debe seguir un procedimiento diferente para subir información a una de las tablas de DynamoDB. Esta tabla en particular es la tabla “DataSet” en la cual se debe subir información desde el DataSet, en el apartado 3 se muestra el esquema y el DataSet con el cual se esta trabajando, la siguiente es la URL donde se encuentra alojado el DataSet:

- [https://github.com/laboratorioAI/PIS\\_20\\_02\\_AGENTES\\_IA/tree/main/Actualizacion\\_2024/Dataset](https://github.com/laboratorioAI/PIS_20_02_AGENTES_IA/tree/main/Actualizacion_2024/Dataset)

El primer paso es el crear o abrir un “NoteBook de SageMaker” como se muestra en el apartado 4.2, en este caso se hará uso de la instancia que ya se había creado y que se ya se encuentra en servicio.

Dentro del entorno de “Jupyter” se debe crear un nuevo NoteBook de Python llamado “DynamoDB”, este procedimiento se muestra en el apartado 4.2 y se muestra en la Figura 103.

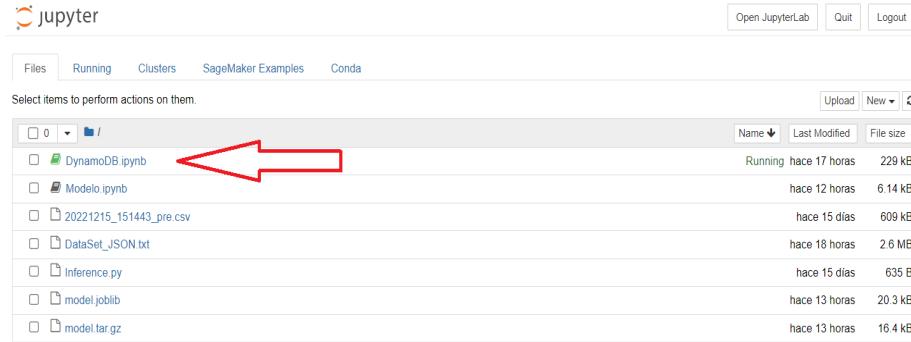


Figura 103: DynamoDB NoteBook.

El siguiente paso en el procedimiento, es convertir el DataSet que se esta utilizando a formato “JSON” para que se pueda agregar a la tabla de DynamoDB, recordar que se puede agregar información a las tablas de DynamoDB de dos maneras: manualmente o en formato “JSON” como se menciono en el apartado 5.6 (Figura 101).

Para transformar la información del DataSet en el formato requerido se puede hacer uso de diferentes herramientas o código, en este caso se uso la herramienta “CSVJSON” proporcionada por “FlatFile”, la siguiente es la URL donde se encuentra alojado el servicio:

■ <https://csvjson.com/>

La Figura 104 muestra los resultados del cambio de formato y las configuraciones que se debe realizar dentro del servicio.

The screenshot shows the "CSV or TSV > JSON" converter interface with the following details:

- Input:** "Or paste your CSV here" with a large text area containing CSV data.
- Settings:**
  - "Separator": "Auto-detect" (selected)
  - "Parse numbers": checked
  - "Parse JSON": checked
  - "Transpose": unchecked
  - "Output": "Array" (selected)
  - "Hash": unchecked
  - "Minify": unchecked
- Output:** A large text area displaying the generated JSON code.
- Buttons:** "Convert", "Clear", "Download", "Save", and "Copy to clipboard".

Figura 104: CSV a JSON.

Una vez realizada la transformación este archivo se debe subir al entorno de Jupyter de AWS para poder tener la información resguardada en un lugar seguro y de fácil acceso. Para subir el archivo solo se debe presionar el botón de “Upload”, buscar el archivo y subirlo. Ver Figuras 105 y Figura 106.



Figura 105: Upload DataSet JSON.

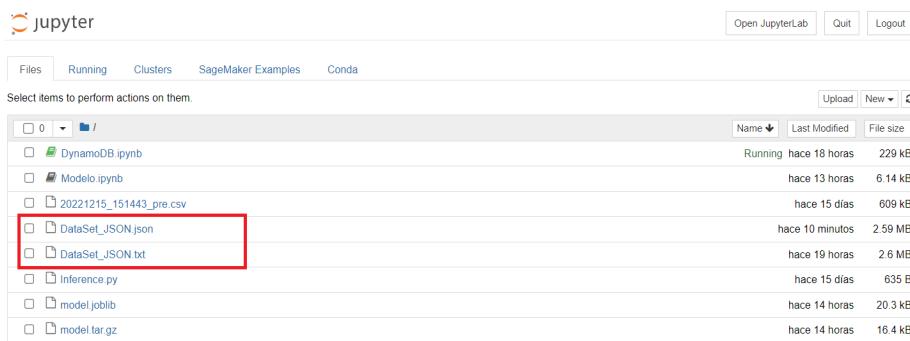


Figura 106: Jupyter DataSet JSON y TXT.

Como se aprecia en la Figura 106 se muestran dos archivos: el primero con extensión “JSON” y el segundo con extensión “TXT”, estos archivos básicamente son lo mismo pero el segundo archivo se encuentra más estilizado para que el entendimiento de la información sea mejor sea mejor. Ver Figura 107.



Figura 107: Diferencia de Estilos.

Con todos los archivos creados y subidos al entorno de Jupyter, corresponde a la programación del código en python. Para ello en el archivo “DynamoDB.ipynb” se realiza el código mostrado en la Figura 108.

```
# Importación de librerías necesarias para realizar consultas en DynamoDB
import boto3

# Definición de la función upload_dynamoDB
def upload_dynamoDB(data):
    # Configuración de las credenciales de AWS
    session = boto3.Session(
        aws_access_key_id='AKIA6GBMNGUTS3GJOPCL7',
        aws_secret_access_key='67z+euHRudmNvt92BHNJZ8KG9QofSvt/kY2W6wSF',
        region_name='us-east-1'
    )

    # Crear una instancia del cliente DynamoDB
    dynamodb = session.resource('dynamodb')
    # Nombre de la Tabla en DynamoDB
    table_name = 'Datos'
    # Crear un objeto en La Tabla
    table = dynamodb.Table(table_name)
    # Subir los datos a DynamoDB
    for item in data:
        id = int(item['id'])
        time = item['time']
        steering_angle = int(item['steering_angle'])
        speed = int(item['speed'])
        rpm = int(item['rpm'])
        acceleration = int(item['acceleration'])
        throttle_position = int(item['throttle_position'])
        engine_temperature = int(item['engine_temperature'])
        system_voltage = int(item['system_voltage'])
        barometric_pressure = int(item['barometric_pressure'])
        distance_travelled = int(item['distance_travelled'])
        distance_travelled_total = int(item['distance_travelled_total'])
        id_vehicle = int(item['id_vehicle'])
        latitude = int(item['latitude'])
        longitude = int(item['longitude'])
        altitude = int(item['altitude'])
        id_driver = int(item['id_driver'])
        heart_rate = int(item['heart_rate'])
        stress = int(item['stress'])
        body_battery = int(item['body_battery'])
        current_weather = int(item['current_weather'])
        current_weather_category = item['current_weather_category']
        has_precipitation = int(item['has_precipitation'])
        is_day_time_category = item['is_day_time_category']
        is_day_time = int(item['is_day_time'])
        temperature = int(item['temperature'])
        real_feel_temperature = int(item['real_feel_temperature'])
        wind_speed = int(item['wind_speed'])
        wind_direction = int(item['wind_direction'])
        relative_humidity = int(item['relative_humidity'])
        visibility = int(item['visibility'])
        uv_index = int(item['uv_index'])
        uv_index_text = item['uv_index_text']
        cloud_cover = int(item['cloud_cover'])
        ceiling = int(item['ceiling'])
        pressure = int(item['pressure'])
        precipitation = int(item['precipitation'])
        accidents_onsite = int(item['accidents_onsite'])
        steering_angle_ = int(item['steering_angle_'])
        speed_vs_steerangle_ = int(item['speed_vs_steerangle_'])
        rpm_ = int(item['rpm_'])
        speed_vs_rpm_ = int(item['speed_vs_rpm_'])
        speed_vs_accidents_onsite_ = int(item['speed_vs_accidents_onsite_'])
        speed_vs_precipitation = int(item['speed_vs_precipitation'])
        accident_rate = int(item['accident_rate'])

        # Ingresando los datos
        table.put_item(
            Item={
                'id': id,
                'time': time,
                'steering_angle': steering_angle,
                'speed': speed,
                'rpm': rpm,
                'acceleration': acceleration,
                'throttle_position': throttle_position,
                'engine_temperature': engine_temperature,
                'system_voltage': system_voltage,
                'barometric_pressure': barometric_pressure,
                'distance_travelled': distance_travelled,
                'distance_travelled_total': distance_travelled_total,
                'id_vehicle': id_vehicle,
                'latitude': latitude,
                'longitude': longitude,
                'altitude': altitude,
                'id_driver': id_driver,
                'heart_rate': heart_rate,
                'stress': stress,
                'body_battery': body_battery,
                'current_weather': current_weather,
                'current_weather_category': current_weather_category,
                'has_precipitation_category': has_precipitation_category,
                'is_day_time_category': is_day_time_category,
                'is_day_time': is_day_time,
                'temperature': temperature,
                'real_feel_temperature': real_feel_temperature,
                'wind_speed': wind_speed,
                'wind_direction': wind_direction,
                'relative_humidity': relative_humidity,
                'visibility': visibility,
                'uv_index': uv_index,
                'uv_index_text': uv_index_text,
                'cloud_cover': cloud_cover,
                'ceiling': ceiling,
                'pressure': pressure,
                'precipitation': precipitation,
                'accidents_onsite': accidents_onsite,
                'steering_angle_': steering_angle_,
                'speed_vs_steerangle_': speed_vs_steerangle_,
                'rpm_': rpm_,
                'speed_vs_rpm_': speed_vs_rpm_,
                'speed_vs_accidents_onsite_': speed_vs_accidents_onsite_,
                'speed_vs_precipitation': speed_vs_precipitation,
                'accident_rate': accident_rate
            }
        )
    print("Subiendo los datos a DynamoDB")

# Llamado a la función creada
data = [
    {"Id":0,"time":"15:14:56","steering_angle":38.7,"speed":0,"rpm":663,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":1,"time":"15:14:57","steering_angle":38.7,"speed":0,"rpm":665,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":2,"time":"15:14:58","steering_angle":38.7,"speed":0,"rpm":667,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":3,"time":"15:14:59","steering_angle":35.5,"speed":0,"rpm":670,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":4,"time":"15:15:00","steering_angle":35.5,"speed":0,"rpm":664,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":5,"time":"15:15:01","steering_angle":35.5,"speed":0,"rpm":663,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":6,"time":"15:15:02","steering_angle":35.5,"speed":0,"rpm":666,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":7,"time":"15:15:03","steering_angle":35.5,"speed":0,"rpm":665,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":8,"time":"15:15:04","steering_angle":35.5,"speed":0,"rpm":681,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":9,"time":"15:15:05","steering_angle":35.5,"speed":0,"rpm":672,"acceleration":0,"throttle_position":12.94117647,"engine":0,"Id":10,"time":"15:15:06","steering_angle":35.5,"speed":0,"rpm":665,"acceleration":0,"throttle_position":12.94117647,"engine":0}
]
upload_dynamoDB(data)
```

Figura 108: Código DynamoDB Jupyter.

El código mostrado en la Figura 108 permite subir tantos datos como el usuario desee, incluso miles de datos al mismo tiempo a la base de datos de DynamoDB. Tras ejecutar el código se puede apreciar que en la tabla “DataSet” se crearon todas las filas. Ver Figura 109.

	<b>Id (Número)</b>	acceleration	accident_rate	accidents_onsite	altitude
0	0	2	21	0	0
1	0	2	21	0	0
2	0	2	21	0	0
3	0	2	21	0	0
4	0	2	21	0	0
5	0	2	21	0	0
6	0	2	21	0	0

Figura 109: Elementos creados en la Tabla DataSet.

## 6. Llamada a la API Gateway desde la APP

Se diseñó e implementó una aplicación móvil para Android basada en Java. La aplicación envía una observación que incluye datos del vehículo, conductor, condiciones climáticas y accidentes de tránsito hacia el REST API implementado por el agente de procesamiento. El REST API recibe como entrada esa información y devuelve como salida el nivel de riesgo de sufrir un accidente de tránsito. La Figura 110 presenta el esquema de infraestructura usado por el agente de respuesta.

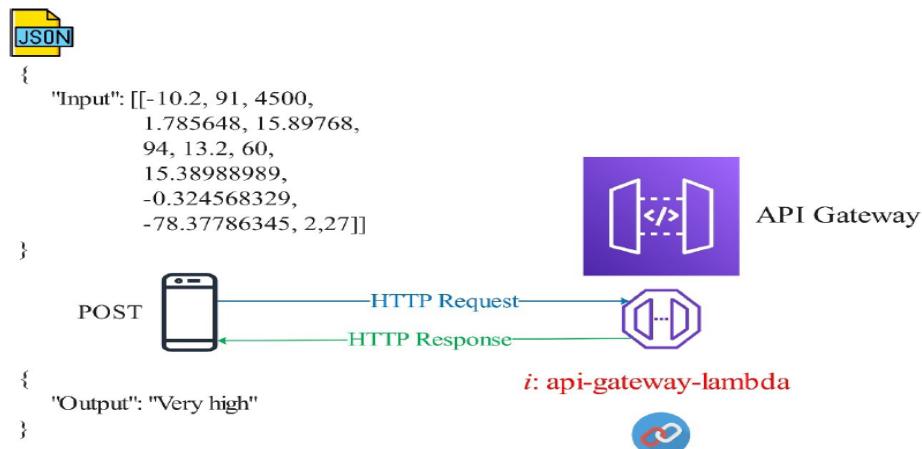


Figura 110: Infraestructura Agente Respuesta.

Con todo el diseño y los servicios implementados se procede a probar el funcionamiento de la APP, para lo cual se debe seguir con el procedimiento establecido en el manual de usuario. El manual de usuario se encuentra en el siguiente URL:

- [https://github.com/laboratorioAI/PIS\\_20\\_02\\_AGENTES\\_IA/tree/main/Actualizacion\\_2024/Manual%20V3.0](https://github.com/laboratorioAI/PIS_20_02_AGENTES_IA/tree/main/Actualizacion_2024/Manual%20V3.0)

Una vez que se ejecuta la aplicación, se aprecia que la APP comienza a mostrar datos sobre el vehículo y las alertas que se haya configurado en el dispositivo. Ver Figura 111.

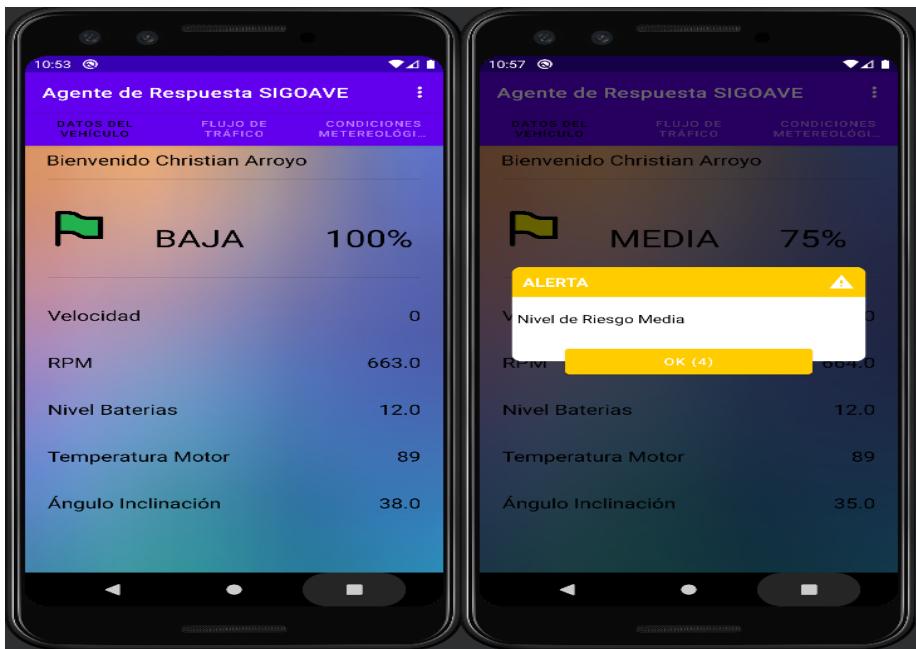


Figura 111: APP Alertas.

Adicionalmente, si la aplicación esta siendo ejecutada desde el entorno de desarrollo de “Android Studio” se puede apreciar como se envían y se reciben las peticiones POST/REST desde la API Gateway. Ver Figura 112.

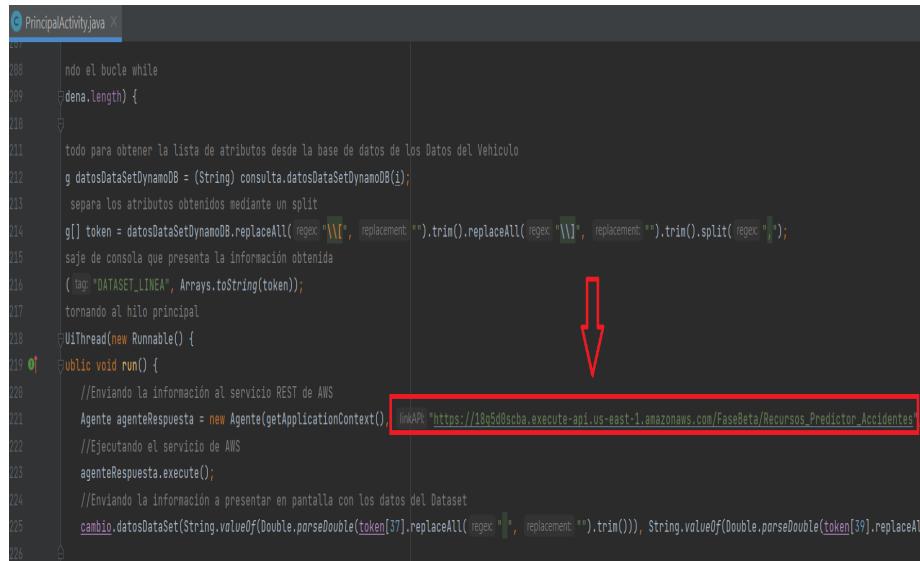
```
2024-02-16 11:05:22.707 6018-6100 com.amazonaws.request com.proyecto.pis_20_02
2024-02-16 11:05:22.707 6018-6100 com.amazonaws.request com.proyecto.pis_20_02
2024-02-16 11:05:23.583 6018-6100 com.amazonaws.request com.proyecto.pis_20_02
2024-02-16 11:05:23.589 6018-6100 DATASET_LINEAR com.proyecto.pis_20_02
2024-02-16 11:05:24.224 6018-6101 RESULTADO API: com.proyecto.pis_20_02
2024-02-16 11:05:24.242 6018-6104 AWS4Signer com.proyecto.pis_20_02

[0] GET/CloudFront 164.54.61.92
[0] Content encoding: null
[0] Done parsing service response
[0] [1, 15:14:57, 38, 0, 645, 0, 12, 89, 12, 18, 0, 2, 1, 8, -78]
[0] {"Output": "2"}
[0] AWS4 Canonical Request: "POST
/
host:dynamodb.us-east-1.amazonaws.com
x-amz-date:20240216T160524Z
x-amz-target:DynamoDB_20120810.Scan

host;x-amz-date;x-amz-target
2fae0a29a2d12daefad88dd8aee59cb76676206b70544ec4285F9b4b"
[0] AWS4 Key Sign: "AWSK-HMAC-SMA256
28240216T160524Z
20240216/us-east-1/dynamodb/aws-request
e8deffe3849c60be977feaf1d3185b2af1a29e71bbd015887802e54b72af4"
[0] Parsing service response JSON
[0] CRC32Checksum = 3591601972
[0] Content encoding: null
[0] Done parsing service response
[1] [{"Media": "2"}]
[1] Media
[1] {"Output": "2"}
```

Figura 112: APP Alertas Logcat.

Se debe recordar cambiar la URL de invocación de la API Gateway para peticiones POST/REST, en el código de la APP, cada que se cree una nueva API. La URL de invocación, como se muestra en la Figura 66 del apartado 4.3, cambiara cada vez que se altere directamente la API Gateway. Ver Figura 113, esto se encuentra en la clase “PrincipalActivity.java”.



```

100     ndo el bucle while
101     dena.length) {
102
103
104     todo para obtener la lista de atributos desde la base de datos de los Datos del Vehículo
105     g datosDataSetDynamoDB = (String) consulta.datosDataSetDynamoDB(1);
106     separa Los atributos obtenidos mediante un split
107     g[] token = datosDataSetDynamoDB.replaceAll( regex "[M]", replacement "").trim().replaceAll( regex "[W]", replacement "").trim().split( regex "[;]");
108     saje de consola que presenta la información obtenida
109     (tag: "DATASET_LINEA", Arrays.toString(token));
110     tornando al hilo principal
111     new Thread(new Runnable() {
112
113         public void run() {
114             //Enviando la información al servicio REST de AWS
115             Agente agenteRespuesta = new Agente(getApplicationContext());
116             linkAPI: "https://18050scha.execute-api.us-east-1.amazonaws.com/FaseBeta/Recursos_Predictor_Accidentes"
117             //Ejecutando el servicio de AWS
118             agenteRespuesta.execute();
119             //Enviando la información a presentar en pantalla con los datos del Dataset
120             cambio_datosDataSet(String.valueOf(Double.parseDouble(token[37]).replaceAll( regex "[", replacement "").trim()), String.valueOf(Double.parseDouble(token[39].replaceAll( regex "[", replacement "").trim())));
121         }
122     }).start();
123
124
125
126

```

Figura 113: Cambio URL APP.

Finalmente, en la Figura 114 se muestra el “Esquema Final del Sistema de Predicción” tras crear, implementar y unir todos los servicios dentro de la APP.

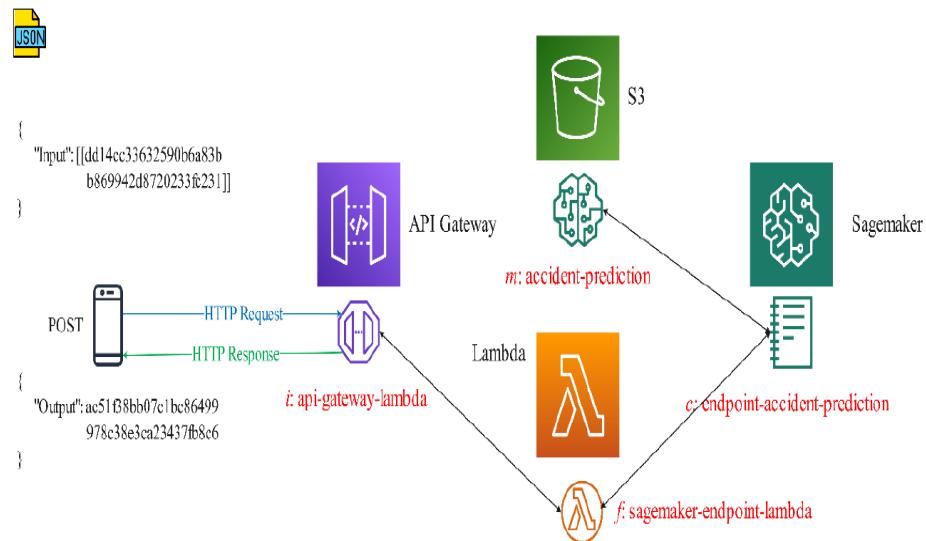


Figura 114: Esquema Final Sistema Predicción.

## 7. Optimización de la APP

La optimización y las mejoras continúan a lo largo del desarrollo. Esto implica optimizar el rendimiento de la aplicación, abordar posibles problemas de seguridad y realizar mejoras en la interfaz de usuario o la funcionalidad según las retroalimentaciones y necesidades del usuario.

### 7.1. SDK de Google Maps

Una de las mejoras implementadas es un sistema para navegación mediante mapas en la cual se registra la ubicación del usuario. Para ello se debe crear un proyecto dentro de “Maps SDK for Android” que se encuentra en la siguiente dirección URL (ver Figura 115):

- <https://developers.google.com/maps/documentation/android-sdk?hl=es-419>

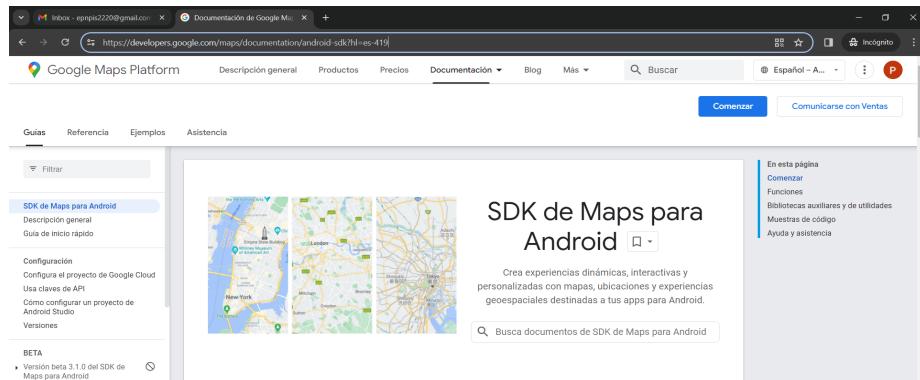


Figura 115: Maps SDK for Android.

Tras presionar el botón “Comenzar”, previamente se debe haber iniciado sesión con una cuenta de GMAIL, se cambiara de interfaz donde se tendrá que crear un nuevo proyecto, ver Figura 116.

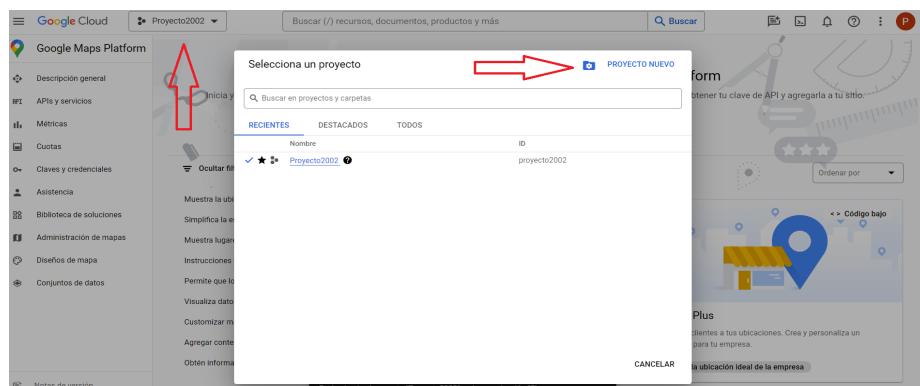


Figura 116: Nuevo Maps SDK.

Al presionar el botón “Proyecto Nuevo” se cambiara de interfaz donde se deberá configurar (Nombre y Organización - opcional) el proyecto como muestra la Figura 117.

### Proyecto nuevo

Tienes 10 projects restantes en tu cuota. Solicita un incremento o borra algunos proyectos. [Más información](#)

[MANAGE QUOTAS](#)

Nombre del proyecto \* \_\_\_\_\_

ProyectoPis2002

ID del proyecto: proyectoPis2002-414517. No se puede cambiar más adelante.  
[EDITAR](#)

Ubicación \* \_\_\_\_\_

Sin organización [EXPLORAR](#)

Organización o carpeta superior

[CREAR](#)   [CANCELAR](#)

Figura 117: Nuevo Proyecto Maps.

AL crear el proyecto se regresara a la interfaz principal donde se apreciara que el proyecto se ha creado, en esta interfaz se debe seleccionar el proyecto creado para empezar a trabajar. Ver Figura 118.

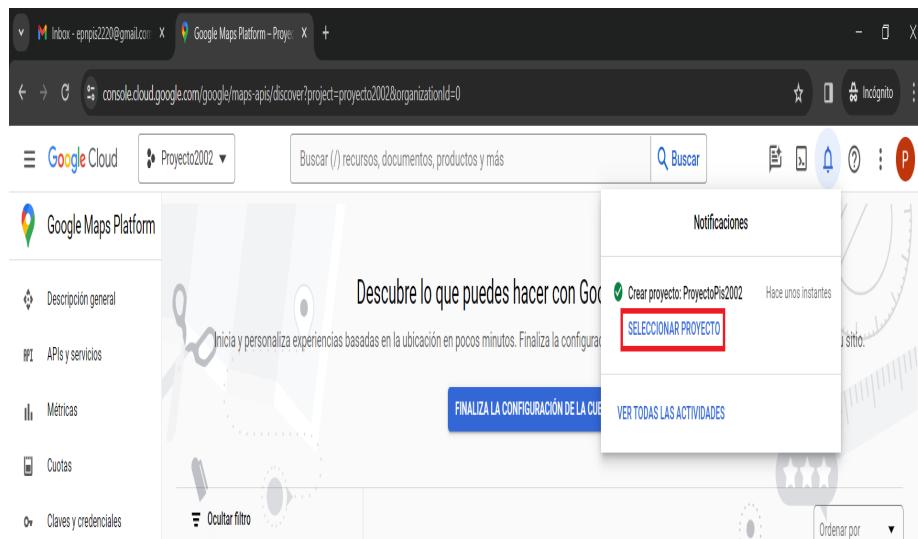


Figura 118: Selección Proyecto.

Después de seleccionar el proyecto en el cual se trabajara, en el panel izquierdo del panel de control se debe seleccionar la opción “API y Servicios” como muestra la Figura 119.



Figura 119: API y Servicios.

En la pantalla de “API y Servicios” se debe terminar de configurar la cuenta, si no se lo ha hecho, como país de origen, cuenta bancaria, etc.; los servicios de “Maps SDK for Android” son gratuitos los primeros 3 meses y si no se supera una determinada cuota.

Tras finalizar la configuración, se mostrara en pantalla que el servicio para mapas se ha creado y se muestra la “Clave API”, ver Figura 120 la cual también debe ser colocada dentro de la APP de Android desarrolla en el archivo llamado “Manifest” como muestra la Figura 121.

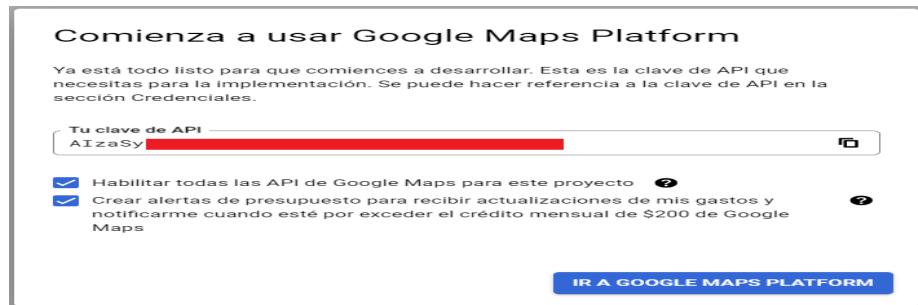


Figura 120: Clave API SDK.

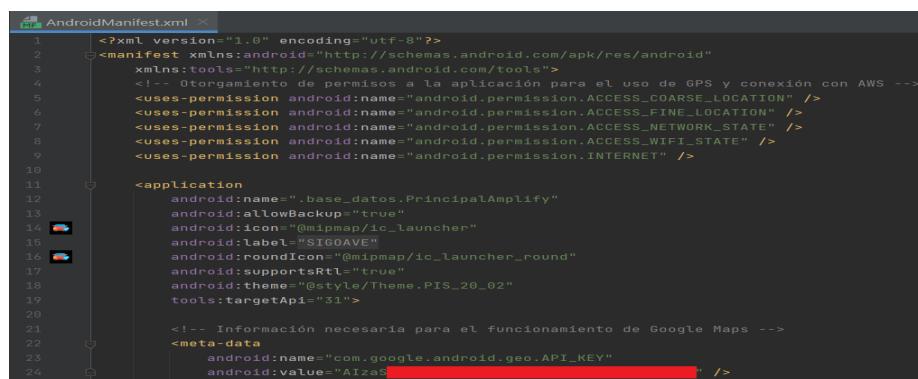


Figura 121: Clave API SDK Android.

Finalmente, en la pantalla de “API y Servicios” se puede observar que la “Maps SDK for Android” se encuentra habilitada, ver Figura 122.

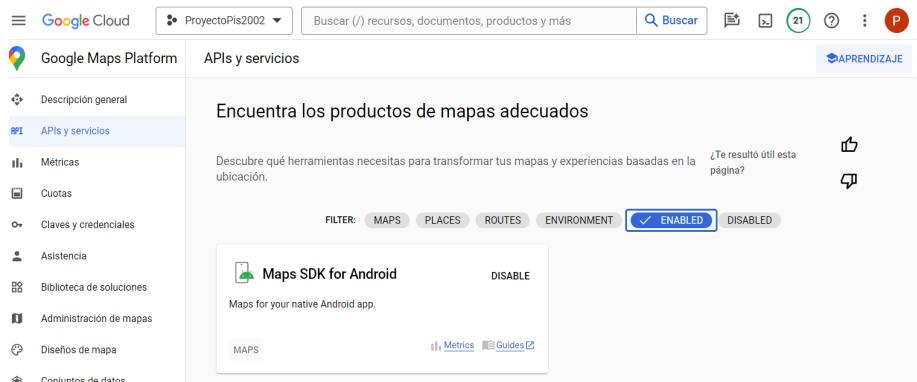


Figura 122: Maps SDK for Android Habilitada.

El ultimo paso es comprobar el funcionamiento dentro de la APP. Ver Figura 123.



Figura 123: Maps SDK Android Funcionamiento APP.

## 8. Baja de Servicios AWS

Para evitar gastos innecesarios mientras la APP no esta en uso, es recomendable dar de baja los servicios de AWS poniendo principal énfasis en los servicios de “SageMaker” que son los que más recursos consume.

Dentro del panel de control de AWS se debe ingresar al panel de control de “SageMaker” en el cual se puede observar cuales son los servicios que se encuentran activos, ver Figura 124.

Figura 124: Servicios de AWS SageMaker Activos.

En esta sección lo que interesa es la opción “Inferencia” que es donde se encuentra los servicios que se van a dar de baja, ver Figura 125.

Figura 125: Servicios de AWS SageMaker a dar de Baja.

En cada una de las opciones de Inferencia: “Modelos”, “Configuraciones de punto de enlace” y “Puntos de conexión” existe un elemento el cual debe ser eliminado, ver Figura 126.

Figura 126: Detalle Servicios a dar de Baja.

Antes de eliminar cada servicio se mostrara un mensaje de alerta. La eliminación de los servicios no se puede deshacer. Si se desea restablecer los servicios, estos tiene que ser creados nuevamente desde cero. Ver Figura 127.

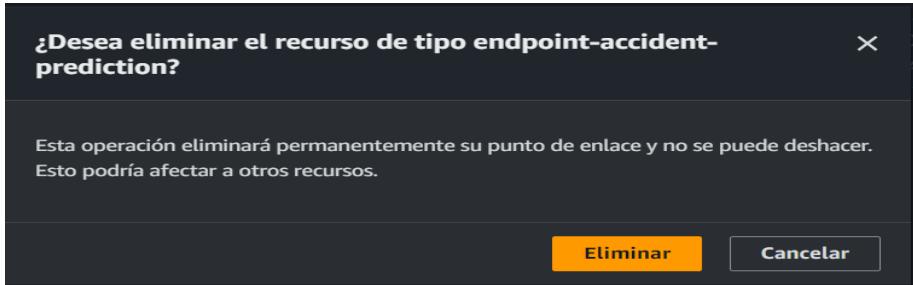


Figura 127: Alerta Eliminación Servicio.

Tras eliminar los servicios, en el panel de control de "SageMaker" se podrá observar que no existen servicios activos. Adicionalmente, se aprecia que se encuentra activo una instancia de "Bloc de Notas". Ver Figura 128, esta no es necesario eliminar el servicio, solo es necesario suspender el servicio.

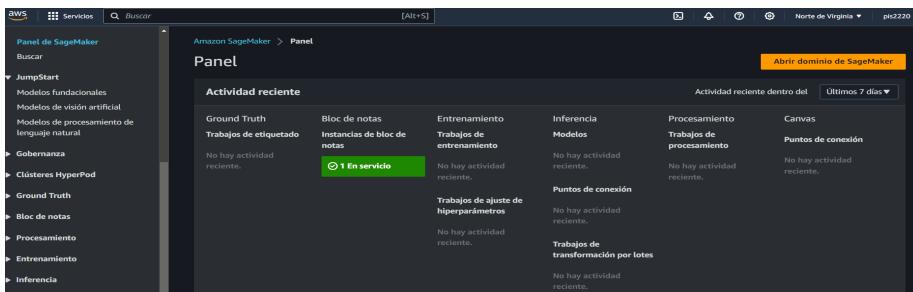


Figura 128: Instancia de Bloc de Notas Activa.

Para suspender el servicio se debe ir a la opción de "Bloc de notas", luego a "Instancias de bloc de notas" lo cual abrirá un panel donde se encuentra la instancia a suspender. En este panel se escoge la instancia a suspender y en la opción "Acciones" se escoge la opción "Detener", ver Figura 129. Este proceso puede tardar varios minutos.



Figura 129: Suspender Bloc de Notas.

Tras suspender la instancia, se puede apreciar en el panel de control de SageMaker que la instancia ya no se encuentra activa, ver Figura 130.

Nombre	Instancia	Hora de creación	Estado	Acciones
ModeloAgenteSagemaker	ml.t2.medium	1/2/2024, 9:36:09	Stopped	Iniciar

Figura 130: Bloc de Notas Suspensos.

Para recrear los servicios eliminados se debe realizar los pasos mostrados en el apartado 4.2 desde la Figura 33 a la Figura 39. Tras esperar un par de minutos para que los servicios se restablezcan se debe continuar con los pasos especificados en el apartado 4.2.1 desde la Figura 51 a la Figura 55.

## 9. Resultados

Una vez completado el desarrollo, se genera un archivo APK (Archivo de Paquete de Android), que es el paquete final de la aplicación. Este archivo contiene todos los recursos necesarios para instalar y ejecutar la aplicación en dispositivos Android. La aplicación se puede distribuir a través de la tienda de aplicaciones de Google Play o mediante otros métodos de distribución, como la instalación directa desde un sitio web.

Los resultados obtenidos tras la ejecución del proyecto son las siguientes:

- Aplicación móvil (agente de adquisición) para recolección y almacenamiento de información proveniente de un escáner OBD2, un receptor GPS, un reloj inteligente, un servicio web y de una base de datos.
- Conjunto de datos de conducción denominado POLIDriving.
- Herramienta basada en servicios de nube (agente de procesamiento) para lectura de datos de conducción, procesamiento de información a través de un modelo de aprendizaje y almacenamiento de resultados en una base de datos.
- Aplicación móvil (agente de respuesta) para consulta y presentación de niveles de riesgo de accidentes a usuarios finales (conductores).
- Esquema de seguridad basado en computación criptográfica para protección de información intercambiada entre agentes.

## 10. Ayudas Técnicas

La siguiente lista de enlaces presentan ayudas a la implementación de los servicios vistos en este documento:

- <https://docs.amplify.aws/android/start/project-setup/create-application/>
- <https://docs.amplify.aws/android/start/project-setup/prerequisites/>
- <https://docs.amplify.aws/android/build-a-backend/auth/set-up-auth/>
- <https://www.youtube.com/watch?v=NtLiuG0lXSA>
- <https://www.youtube.com/watch?v=s8DiT2QLyZc>
- <https://www.youtube.com/watch?v=ThQA90KRu5Y>
- [https://www.youtube.com/watch?v=\\_RNl08Lg9Ww](https://www.youtube.com/watch?v=_RNl08Lg9Ww)
- <https://www.youtube.com/watch?v=xb0i5GTJdRk>
- <https://www.youtube.com/watch?v=pOKPQ8rYe6g>

## Referencias

- [AWS, 2023a] AWS (2023a). Amazon api gateway. Disponible en: <https://aws.amazon.com/es/api-gateway/>. Ultimo Acceso: 16 de febrero de 2024.
- [AWS, 2023b] AWS (2023b). Amazon cognito. Disponible en: <https://aws.amazon.com/es/cognito/>. Ultimo Acceso: 16 de febrero de 2024.
- [AWS, 2023c] AWS (2023c). Amazon dynamodb. Disponible en: <https://aws.amazon.com/es/dynamodb/>. Ultimo Acceso: 16 de febrero de 2024.
- [AWS, 2023d] AWS (2023d). Aws identity and access management. Disponible en: <https://aws.amazon.com/es/iam/>. Ultimo Acceso: 16 de febrero de 2024.
- [AWS, 2023e] AWS (2023e). Qué es amazon sagemaker. Disponible en: [https://docs.aws.amazon.com/es\\_es/sagemaker/latest/dg/whatis.html](https://docs.aws.amazon.com/es_es/sagemaker/latest/dg/whatis.html). Ultimo Acceso: 16 de febrero de 2024.
- [AWS, 2023f] AWS (2023f). Qué es aws lambda. Disponible en: [https://docs.aws.amazon.com/es\\_es/lambda/latest/dg/welcome.html](https://docs.aws.amazon.com/es_es/lambda/latest/dg/welcome.html). Ultimo Acceso: 16 de febrero de 2024.
- [AWS, 023s] AWS (2023s). Aws amplify hosting. Disponible en: <https://aws.amazon.com/es/amplify/hosting/>. Ultimo Acceso: 16 de febrero de 2024.
- [JupyterLab, 2024] JupyterLab (2024). Jupyterlab: A next-generation notebook interface. Disponible en: <https://jupyter.org/>. Ultimo Acceso: 16 de febrero de 2024.

[Marcillo et al., 2022] Marcillo, P., Valdivieso Caraguay, Á. L., and Hernández-Álvarez, M. (2022). A systematic literature review of learning-based traffic accident prediction models based on heterogeneous sources. *Applied Sciences*, 12(9):4529.

[Studio, 2024] Studio, A. (2024). Android studio. Disponible en: <https://developer.android.com/studio?hl=es-419>. Ultimo Acceso: 16 de febrero de 2024.