**LABORATORIO OBLIGATORIO: Aplicación Web de Publicaciones Seguras con AWS Cognito y DynamoDB**

**Contexto: SecurePost Inc.** es una empresa emergente de tecnología que busca ofrecer un servicio de **publicaciones personales seguras en la nube**.  
Su idea es permitir que cada usuario pueda almacenar de manera privada sus notas, recordatorios o ideas importantes en un espacio totalmente seguro, donde **solo ellos** puedan acceder a sus datos.

Para lanzar su primer prototipo, **SecurePost Inc.** ha decidido construir una **aplicación web** basada en **servicios serverless** de AWS, minimizando costos y tiempos de desarrollo.

La aplicación debe garantizar:

* **Seguridad y privacidad de los usuarios.**
* **Escalabilidad automática** sin necesidad de gestionar servidores.
* **Simplicidad** para el usuario final: registro rápido, autenticación, y gestión de sus publicaciones.

Por motivos de agilidad y costo, la empresa ha decidido usar:

* **Amazon Cognito** para gestionar usuarios y autenticar identidades.
* **Amazon DynamoDB** como base de datos NoSQL para almacenar publicaciones de forma eficiente.

SecurePost Inc. ha contactado a tu equipo de desarrolladores para diseñar y construir el primer prototipo funcional.

**Requerimientos principales del cliente:**

Los usuarios deben poder **registrarse** con su correo electrónico y **verificar** su cuenta.

 Los usuarios deben poder **iniciar sesión** y **mantenerse autenticados** de forma segura.

 Cada usuario debe poder:

* Crear nuevas publicaciones (título y contenido).
* Ver solamente sus propias publicaciones.
* Eliminar sus propias publicaciones.

 No debe haber posibilidad de que un usuario vea o altere las publicaciones de otros usuarios.

 El sistema debe ser **sin servidores** (serverless), usando servicios administrados de AWS.

 Las credenciales de AWS deben ser **temporales y seguras** (a través de Cognito Identity Pool).

**Alcance del primer prototipo (MVP):**

* Frontend funcional accesible vía navegador.
* Registro y autenticación usando Cognito User Pool.
* Obtención de credenciales de AWS vía Cognito Identity Pool.
* Operaciones de lectura, escritura y borrado en DynamoDB filtradas por usuario autenticado.

**Branding:**

SecurePost Inc. tiene un estilo simple y profesional:

* Colores: Azul (#0055FF) y Blanco (#FFFFFF).
* Logo: Un pequeño icono de un candado.

**Servicios de AWS involucrados:**

* Cognito User Pool: Gestión de usuarios (registro, inicio de sesión).
* Cognito Identity Pool: Obtención de credenciales temporales de AWS para acceder a otros servicios.
* Amazon DynamoDB: Almacenamiento y gestión de las publicaciones.

**Modelo de datos** para la tabla **Publicaciones** de DynamoDb:

| **Nombre del atributo** | **Tipo de dato** | **Descripción** |
| --- | --- | --- |
| **UserId** | String (Partition Key) | Identificador único del usuario autenticado (viene de Cognito como sub). |
| **PublicacionId** | String (Sort Key) | Identificador único de la publicación (UUID generado al crear la publicación). |
| **Titulo** | String | Título de la publicación. |
| **Contenido** | String | Contenido de la publicación (texto largo). |
| **FechaCreacion** | String (ISO 8601) | Fecha de creación de la publicación en formato ISO 8601 (yyyy-MM-dd'T'HH:mm:ss'Z'). |

Definición de claves:

**Partition key (UserId):** Este campo será utilizado para particionar la tabla, asegurando que las publicaciones estén asociadas a cada usuario por separado. El UserId provendrá del atributo sub de Cognito (un identificador único para cada usuario).

**Sort key (PublicacionId):** Un identificador único para cada publicación, que podría ser un UUID generado al momento de la creación de la publicación.

Ejemplo de **política IAM** para acceso a DynamoDB (para usuarios autenticados):

Como la aplicación debe garantizar que los usuarios solo puedan acceder a sus propias publicaciones, es importante configurar una política IAM que imponga esta restricción. Aquí tienes un ejemplo de política que podrías asociar con el **Identity Pool**:

