Task 4.2 – Serverless Functions & Authentication

1. Real-World Example of Serverless Usage

Product: Netflix

How Netflix Uses Serverless:
Netflix leverages AWS Lambda to manage real-time encoding, monitor user activity, and automate backup and recovery processes. For instance, when a user starts watching a video, Lambda functions are triggered to log data, monitor performance, and scale content delivery based on load.

Why It's Useful:
The serverless architecture allows Netflix to automatically scale based on demand (especially important during high-traffic periods), reduce infrastructure maintenance, and deploy updates faster. This results in better performance and lower operational costs.

2. How I Will Use Serverless Functions

In the Meet App, I will use serverless functions hosted on AWS Lambda to handle the OAuth 2.0 authorization process. These functions will request and receive access tokens from the Google Authorization server, allowing the React frontend to access the Google Calendar API securely. This approach keeps sensitive operations away from the client-side code.

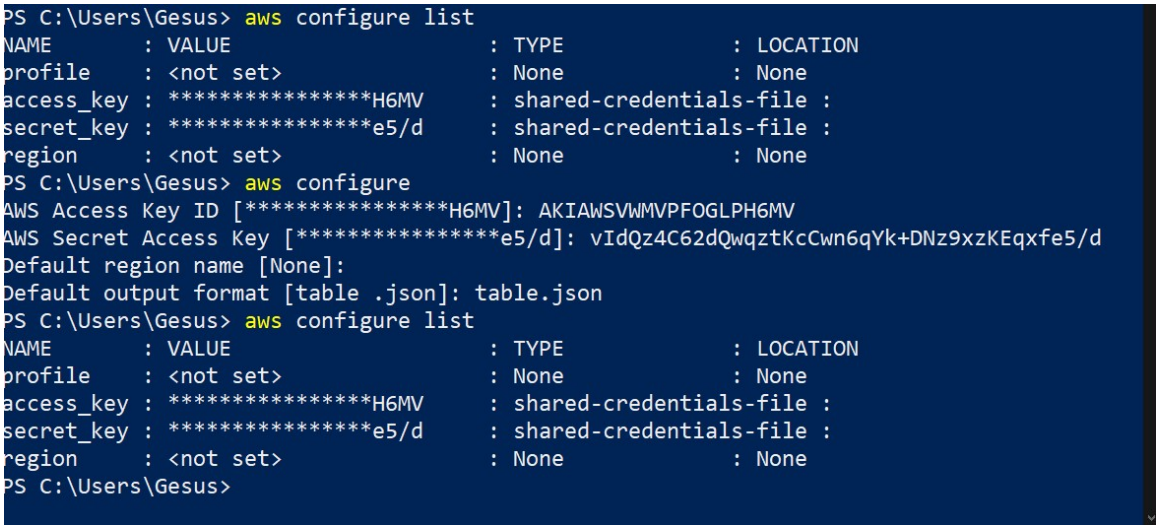
3. Architectural Diagram of the Meet App

The diagram follows the serverless model:
- React Frontend hosted on Vercel
- AWS Lambda handles server-side logic (e.g., token exchange)
- Google Calendar API as external backend service

[User] --> [React App (Vercel)] -- 1. Request token --> [AWS Lambda Function - Token Handler] -- 2. Authenticate user (OAuth flow) --> [Google OAuth Server] -- 3. Return access token --> [AWS Lambda Function] -- 4. Send token to frontend --> [React App (Vercel)] -- 5. Fetch events using token --> [Google Calendar API] -- 6. Return list of events --> [React App

shows events to user]

4. Screenshots



- Screenshot of AWS CLI configuration after running `aws configure`