**Team Cybervision**

This document includes the instructions (prompts) gave to the SecureGPT to generate the code for the AI agents. These prompts guide the agents through each step; from getting data from Amazon S3, generating JSON code, and creating the dashboard.

**SecureGPT Agent prompt**

‘*Generate a Python AWS Lambda function that integrates with Amazon S3 and Secure GPT. The function should:*

*List CSV files from an S3 bucket and retrieve column names and sample rows from each dataset.*

*Use the downloaded SSL certificate from another S3 bucket to securely communicate with SecureGPT.*

*Take a user query, classify the user as technical or nontechnical, structure the query using an AI assistant prompt, and send it to Secure GPT via a secure API request.*

*Extract and parse JSON-formatted responses from Secure GPT, ensuring they adhere to a predefined format wrapped between BEGIN\_JSON and END\_JSON.*

*Return the final structured response as JSON, providing insights and visualizations based on the datasets.*

*The code should handle errors gracefully, including failures in S3 operations, JSON parsing issues, API errors, retries for transient failures, and missing input scenarios.*

*Make changes as updates have been made to the code, including user role classification, auto-fixing invalid chart columns, avoiding unsupported chart types, removing SQL-style expressions, and cleaning duplicate charts.* ‘

**Dashboard rendering agent prompt**

*‘Write an AWS Lambda function in Python that loads dashboard chart data from an input payload, retrieves matching datasets from an S3 bucket, and generates interactive line, bar, and pie charts using Plotly based on chart instructions.*

*Handle grouped time series, category filtering, and chart formatting, applying a custom color palette.*

*Invoke a separate Lambda function named SummaryAgent to generate a natural language summary of the dashboard and inject it into the final HTML.*

*Save the complete HTML dashboard to a temporary file, upload it to an S3 output bucket, and return the public URL. Also log the input event to S3 for traceability.*

*Include support for classifying category types using regex and keywords, parsing time ranges from user queries, removing duplicate titles, and auto-adjusting chart frequency.* ‘

**Summary Agent prompt**

*‘Write an AWS Lambda function in Python that interacts with the SecureGPT and the function should:*

*Extract the user query and HTML dashboard content from the Lambda event input.*

*Clean the HTML content by removing scripts, styles, and tags to prepare it for processing.*

*Classify the user’s role as technical or non-technical based on the user query.*

*Construct a prompt for SecureGPT to generate a professional executive-level summary, focusing on cybersecurity threats, findings, risk assessment, and recommendations.*

*Use an SSL certificate stored in S3 to securely communicate with the SecureGPT API.*

*Handle errors gracefully for missing parameters, unsuccessful API responses, and issues during certificate retrieval.*

*Parse the response from Secure GPT and return the formatted executive summary in the JSON response.*

*If Secure GPT fails, return an error message with the failure details in the Lambda response.’*

**Pipeline agent prompt**

‘*Write an AWS Lambda function in Python and the function should:*

*Extract user\_query from the Lambda event input, handling multiple possible formats and keys.*

*Call the SecureGPTAgent Lambda function up to three times to obtain valid dashboard\_data.*

*Parse and validate the response from Secure GPT, checking for the presence of dashboard\_data.*

*Invoke the DashboardAgent Lambda with the user query and dashboard data to generate a dashboard.*

*Extract the dashboard\_url from the DashboardAgent response and parse the S3 bucket and key.*

*Retrieve the HTML dashboard content from the S3 bucket using the parsed URL.*

*Return the HTML content as the response; optionally, include the dashboard URL.*

*Handle and report errors gracefully for missing inputs, failed Lambda invocations, and S3 issues.’*