**ASDA**

1- Fetching browser cookies data -

Summary

To fetch cookies in a web application:

Client-Side JavaScript: Use document.cookie to access and manage cookies-

// Function to get the value of a specific cookie by name

function getCookie(name) {

let value = `; ${document.cookie}`;

let parts = value.split(`; ${name}=`);

if (parts.length === 2) return parts.pop().split(';').shift();

}

// Example: Fetching a cookie named 'token1'

let token1 = getCookie('token1');

console.log(token1);

Server-Side Code: Retrieve cookies from request headers in server environments.

Security: Handle cookies securely and follow best practices to avoid vulnerabilities.

2- We have to track the requests in asda -

npm install applicationinsights

const appInsights = require('applicationinsights');

appInsights.setup('YOUR\_INSTRUMENTATION\_KEY').start();

const appInsights = require('applicationinsights').defaultClient;

appInsights.trackRequest({name: 'GET /api/data', url: '/api/data', duration: 500, resultCode: '200', success: true});

3- AWS lambda - dt

AWS CloudWatch: CloudWatch Logs can capture detailed logs from your Lambda function, including request payloads and responses.

AWS X-Ray helps you trace and analyze requests as they travel through your Lambda function and other AWS services.

AWS CloudTrail provides a record of API calls made in your AWS account, which includes Lambda function invocations.

4- Azure function app to apm entity after that we can use dt

Most APM tools provide specific instructions for integrating with Azure Functions. Typically, this involves installing a monitoring agent or SDK and configuring it with your application.

To integrate New Relic with a .NET Azure Function App, you need to set up the New Relic .NET agent. This involves installing the agent, configuring it, and ensuring it's properly deployed with your function app. Here’s a step-by-step guide on how to do this:

1. Obtain New Relic Account and License Key

Create a New Relic Account:

Sign up for a New Relic account if you don't have one already.

Get Your License Key:

Log in to New Relic.

Navigate to Account settings > API keys to obtain your New Relic License Key.

2. Install New Relic .NET Agent

a. Install the New Relic .NET Agent

Download the New Relic .NET Agent:

Go to the New Relic Downloads page and download the latest version of the .NET Agent.

Install the Agent:

Run the installer and follow the instructions. The installer will place the New Relic .NET Agent files in a default location (e.g., C:\Program Files\New Relic\New Relic .NET Agent).

b. Configure the Agent

Create a newrelic.config File:

The newrelic.config file contains the configuration settings for the New Relic agent. You can either use the default configuration provided by the installer or create a custom one.

Place the newrelic.config file in the root directory of your Azure Function App project.

Here's a sample newrelic.config file:

<configuration>

<agent>

<!-- License Key for New Relic -->

<licenseKey>Your\_New\_Relic\_License\_Key\_Here</licenseKey>

<!-- Application Name -->

<applicationName>Your\_Application\_Name\_Here</applicationName>

<!-- Other configuration options -->

</agent>

</configuration>

Replace Your\_New\_Relic\_License\_Key\_Here with your actual New Relic License Key and Your\_Application\_Name\_Here with a meaningful name for your application.

Update host.json to Include newrelic.config:Ensure the newrelic.config file is included in the deployment package for your Azure Function App. It should be in the root directory of your function app where the host.json file is located.

3. Instrument Your Azure Function App

Since Azure Functions run in a sandboxed environment, ensure that your New Relic agent is properly instrumented for the runtime.

Modify the host.json File (If Required):

Azure Functions use host.json to configure the function runtime. Ensure it includes any necessary settings for telemetry or logging if required by New Relic.

json

Copy code

{

"version": "2.0",

"logging": {

"logLevel": {

"default": "Information"

}

}

}

4. Deploy Your Azure Function App

Deploy the Function App:

Use your preferred method for deploying Azure Functions (Azure CLI, Visual Studio, GitHub Actions, etc.) to ensure the newrelic.config file and the New Relic .NET agent files are included in the deployment.

Verify Deployment:

After deployment, verify that the New Relic agent is functioning by checking the New Relic dashboard for your application’s data.

5. Monitor and Verify

Check New Relic Dashboard:

Log in to the New Relic dashboard and navigate to the APM section.

Look for your application and verify that performance data, such as response times, error rates, and throughput, is being reported.

Set Up Alerts and Notifications:

Configure alerts in New Relic to monitor for anomalies or performance issues in your application.

Additional Tips

Testing Locally: Before deploying to Azure, test your application locally with New Relic to ensure the agent is working correctly.

Agent Documentation: Refer to New Relic’s official .NET agent documentation for advanced configurations and troubleshooting.

Azure Function App Settings: Ensure that your Azure Function App settings do not conflict with the New Relic agent settings. Verify that all required environment variables and configurations are correctly set.

By following these steps, you can integrate New Relic with your .NET Azure Function App and leverage New Relic’s APM features to monitor and optimize application performance.

5- Correlate all services-

1. Generate and Pass Correlation IDs:

When a user initiates a login request, generate a unique correlation ID (UUID) and attach it as a header (e.g., X-Correlation-ID) in every subsequent request across all services. This allows you to trace the entire flow of the login process.

1. Use structured logging in each service to log important events, including the correlation ID. This will make it easier to filter and analyze logs later.

{

"correlationId": "12345",

"event": "login\_request",

"timestamp": "2024-09-16T12:00:00Z",

"service": "Akamai",

"details": "User initiated login"

}

1. Correlation in Cloudflare and Akamai

Create Routing Rules with Correlation ID:

In Cloudflare and Akamai, create rules that log and forward the correlation ID along with requests. Ensure that all relevant headers (including X-Correlation-ID) are passed through to the backend services.