**SERILOG IMPLEMENTATION**

Serilog is a logging library for .NET and C# that allows for more detailed and structured logging than the default .NET logging library. Serilog can be used to log information about application events, errors, and performance metrics. This information can be used to troubleshoot issues with an application or to monitor its performance over time.

Serilog has a few benefits for C# developers, including:

* Simple to configure
* Flexible
* Supports structured logging

**Implementation of Serilog Implementation:**

**Packages:**

Newtonsoft.Json

Serilog.AspNetCore

**In SerilogMiddleware.cs:**

public class SerilogMiddleware

{

public RequestDelegate \_requestDelegate;

private readonly ILogger<SerilogMiddleware> \_logger;

public SerilogMiddleware(RequestDelegate requestDelegate, ILogger<SerilogMiddleware> logger)

{

\_requestDelegate = requestDelegate;

\_logger = logger;

}

public async Task Invoke(HttpContext context)

{

try

{

await \_requestDelegate(context);

}

catch (Exception ex)

{

await HandleException(context, ex);

}

}

private Task HandleException(HttpContext context, Exception ex)

{

\_logger.LogError(ex.ToString());

var errorMessageObject = new { Message = ex.Message, Code = "system\_error" };

var errorMessage = JsonConvert.SerializeObject(errorMessageObject);

if (errorMessage.Contains("400"))

{

context.Response.ContentType = "application/json";

context.Response.StatusCode = (int)HttpStatusCode.BadRequest;

}

if(errorMessage.Contains("401"))

{

context.Response.ContentType = "application/json";

context.Response.StatusCode = (int)HttpStatusCode.Unauthorized;

}

if (errorMessage.Contains("404"))

{

context.Response.ContentType = "application/json";

context.Response.StatusCode = (int)HttpStatusCode.NotFound;

}

if (errorMessage.Contains("405"))

{

context.Response.ContentType = "application/json";

context.Response.StatusCode = (int)HttpStatusCode.MethodNotAllowed;

}

if (errorMessage.Contains("500"))

{

context.Response.ContentType = "application/json";

context.Response.StatusCode = (int)HttpStatusCode.InternalServerError;

}

\_logger.LogError(errorMessage);

return context.Response.WriteAsync(errorMessage);

}

}

**In Program.cs:**

//implementing Serilog

app.UseMiddleware(typeof(SerilogMiddleware));

**appsettings.json:**

"Logging": {

"Region": "us-east-1",

"LogGroup": "Cloud watch group",

"IncludeLogLevel": true,

"IncludeCategory": true,

"IncludeNewline": true,

"IncludeException": true,

"IncludeEventId": false,

"IncludeScopes": false,

"AccessKey": "Some acccess key",

"AccessId": "Some acccess id",

"LogLevel": {

"Default": "Information",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}