APPLICATION

This application can be used to maintain the details of employees with the basic operations of list, create, update and delete employees.

Along with this provided the option to generate DB lock (for SQL server option) and 400 errors in the index page.

TECHNOLOGY STACK

The application is developed in ASP.NET MVC web application, with Azure SQL and COSMOS DB in the back-end.

ARTCHITECTURAL LAYERS

1. View: The pages of the application, which enables user to manage the employee(s).
2. Controller: The acting part of the application which reacts to the action from Model and acts upon. This contains methods for employee management. The application can connect to both COSMOS and SQL (in Azure), provided with correct configurations.
3. Modal: The DB layer of the application. This contains methods for calling the corresponding DB required, based on the actions from View.

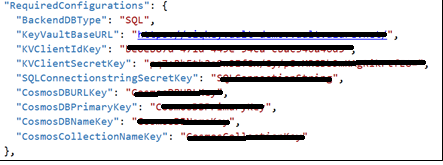
APPLICATION INSIGHTS

The logs of application including requests, exceptions etc., has been captured in the Azure application insights. The configuration of application insight is covered in detail on the later part of this document.

Please note, the application insights will not be recorded for 400 error.

CODE CONFIGURATION

1. The application is configured to use Azure key-vault to read the connection strings for SQL and COSMOS DB. The keys are available in “..LabWebApplication\appSettings.json” under “RequiredConfigurations” section.



KeyVaultBaseURL : URL of the key vault created in your Azure subscription, which will be used to store the secrets being used in the application.

KVClientIdKey & KVClientSecretKey: The client details (application id & secret) which is authorized to access the Key vault client created

SQLConnectionStringSecretKey: Name (or key) of the secret which is holding the connection string of Azure SQL DB created

COSMOSDBURLKey, COSMOSDBPrimaryKey, COSMOSDBNameKey, COSMOSCollectionNameKey: The details of the COSMOS DB you created.

1. The instrumentation key of application insight is available at “..LabWebApplication\Properties\launchSettings.json” under “profiles”. The same can be modified with the instrumentation key of your choice (in your subscription).

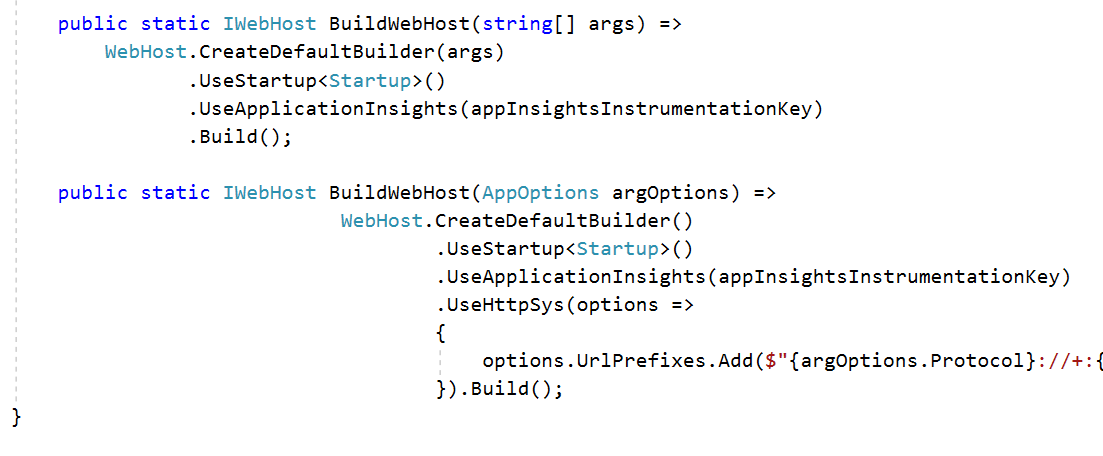


1. In the above section there is another environment variable “BACKEND\_DB”, which decides the DB to be connected. If it’s SQL, SQL details given in the appSettings.json (given in the first point) will be used and the details of COSMOS will be used in case BACKEND\_DB is “COSMOS”

APPLICATION INSIGHTS - IMPLEMENTATION

The following steps are incorporated in this project, to enable application insights.

1. Added the application insight instrumentation key under “environmentVariables” section as shown above. This is added under environment variables, to make sure one can pass this value during deployment in any other platform other than in web server directly. Examples: Service fabric, Cluster etc.,
2. Installed “Microsoft.ApplicationInsights.AspNetCore” in the LabWebApplication project
3. Added “UseApplicationInsights” method in Program.cs for web host. The instrumentation key will be read from environment variable



1. Added the highlighted part below in the “\_Layout.csthml” under Views folder, which is the page loaded on each request of any page. This make sure logging of each requests under “Request” in application insight.

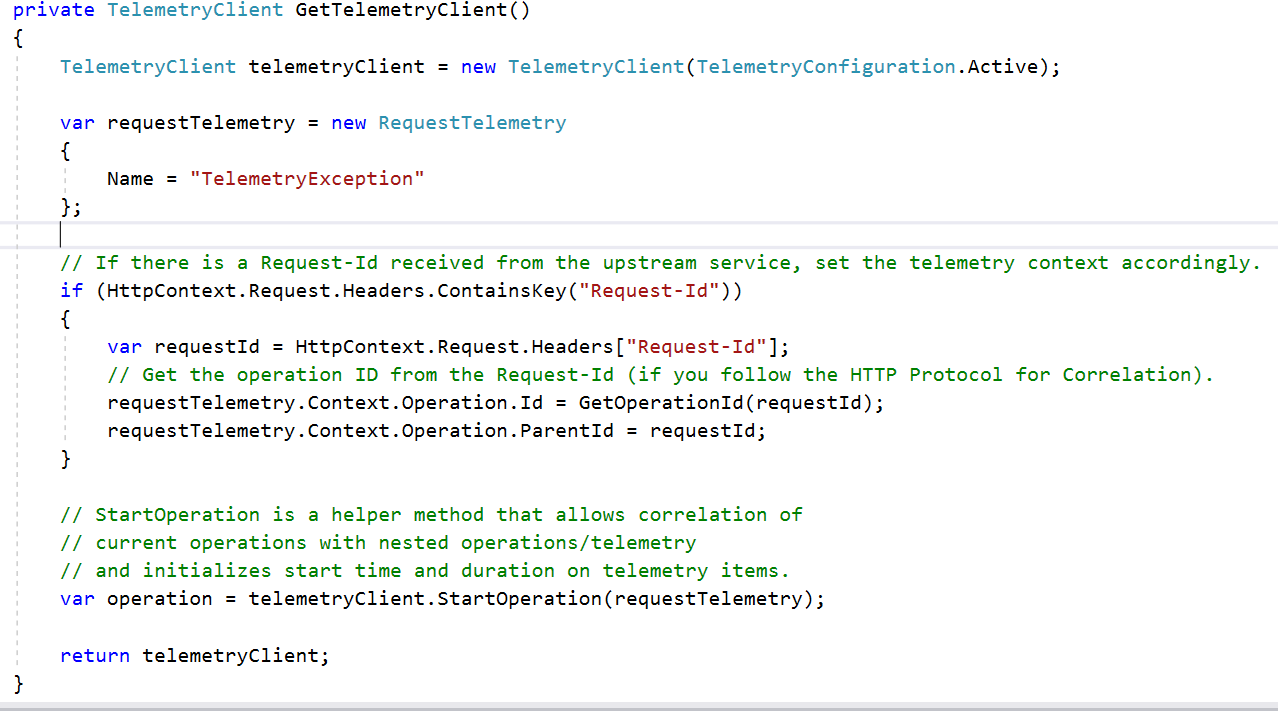


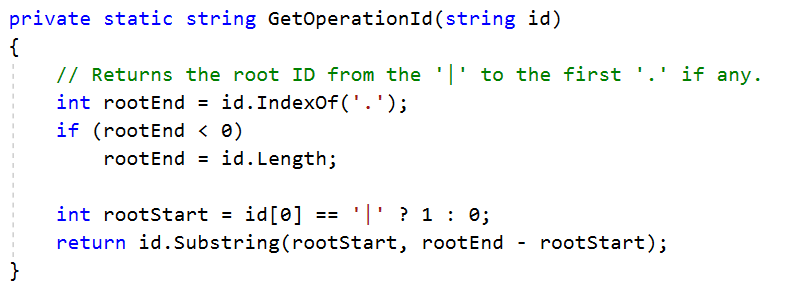
The above implementation takes care of following insights to be logged automatically in the application insight configured

1. Request
2. Exceptions, if occurred
3. Traces
4. Dependencies
5. Browser Timings

In this project we need to generate exception which causes DB lock. This has been implemented for SQL server. For this we need to have custom exceptions logged.

We have introduced the below code snippet in EmployeesController.cs file to get the telemetry client with “Operation Id”.





This telemetry client being used in “GenerateLock” method (under EmployeeController.cs), so that the custom exception will also have the same operation-id as the request.

This will help in inter-linking various application insights like request, exception

