**Scan event exercise:**

**Indika Kamal**

**Requirements**

1. The application must consume from a scan event API and keep a track of the last event that was fetched so that it can continue where it left off if the service is stopped and started.

API is not created, therefore to consume data for the application, the given Json file has been used by reading the file.

In order to find where it left of keep track in a text file. Caching also possible, but in case any issue on server, like restarting server caching data will lose. Also saving in DB table another option, but it may affect performance, and unnecessary DB calls.

1. The application should persist the scan event data in such a way that the following information could later be fetched easily;
2. The most recent scan event against a parcel, specifically only values from fields; EventId, ParcelId, Type, CreatedDateTimeUtc, StatusCode, RunId are required.

All the cleaned-up data save in table level

1. DateTimes indicating when a parcel has been;
   1. Picked up (a scan event of Type ‘PICKUP’ has occurred).
   2. Delivered (a scan event of Type ‘DELIVERY’ has occurred).
2. The application should be fault tolerant and resilient (e.g. handle new event types, malformed data).

Fluent Validation has been implemented.

Load balancing servers can use to for resilient and also another data center option (passive and active)

1. The application should contain appropriate logging.

NLog has been implemented

Assumptions

1. List any assumptions you have made. If anything in these instructions is not clear, write down an assumption that captures this.
   1. API send data in descending order based on most recent records
   2. StatusCode Is not given in Json file, therefore validation is not added for StatusCode property

Improvements

1. List any improvements that could be made to the application you have built including what kind of things could be done to productionise this application.
   1. REST API can use to query data using query string (API URL as per given format)
2. Describe what kind of changes or additions could be performed to enable another worker application downstream to also perform actions against the same scan events processed by this application. Describe what the high-level architecture of this overall system could look like.
   1. Clean architecture with repository pattern.

Used Libraries:

* FluentValidation for validation
* NetwonSoft.Json – Json serialization/deserialization
* NLog – for logging

Database

* Sqlite

Assumptions:

* Assume API provides data in descending order based on EventID

Console application does not have dependency Injection by default

Should not receive Events with blank entry,

* Validation is required

Improvements:

* Use Clean architecture