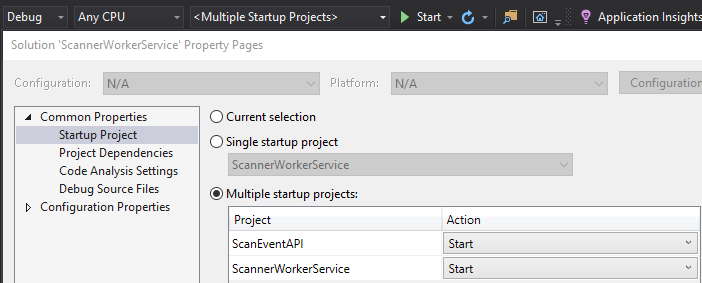
# Introduction:

There are two applications are developed for this exercise:

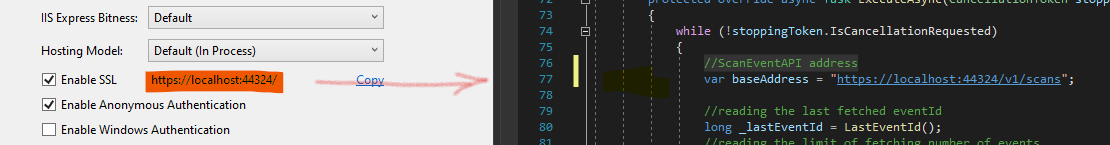
1. **ScannerWorkerService**: The expected worker solution
2. **ScanEventApi**: a supporting API to provide sample data

Both the application developed with C# .Net Core 3.1 in Visual Studio 2019. Considering the ScanEventApi needs to be running before ScannerWorkerService, I have configured Multiple Startup Projects. So it can feed the ScannerWorkerService application. Once after download please make sure the multiple startup setting remains like



# To-do list before testing:

1. Need to have .Net Core 3.1 framework installed
2. To initialise the database; Supplied a folder called 001\_DatabaseCreation to configure a database named ScannerWSDB. Under the folder there are two scripts, first one to create the Database and the tables. And the second script is to produce the initial values/data on which the ScannerWorkerService depends on.
3. Need to update the server name in connectionstring under the appsettings.json file for ScannerWorkerService. At the moment it is written as LT-MASUD\\SQLEXPRESS
4. Make sure the baseAddress is pointing to the right ScanEventAPI address



# Assumptions List:

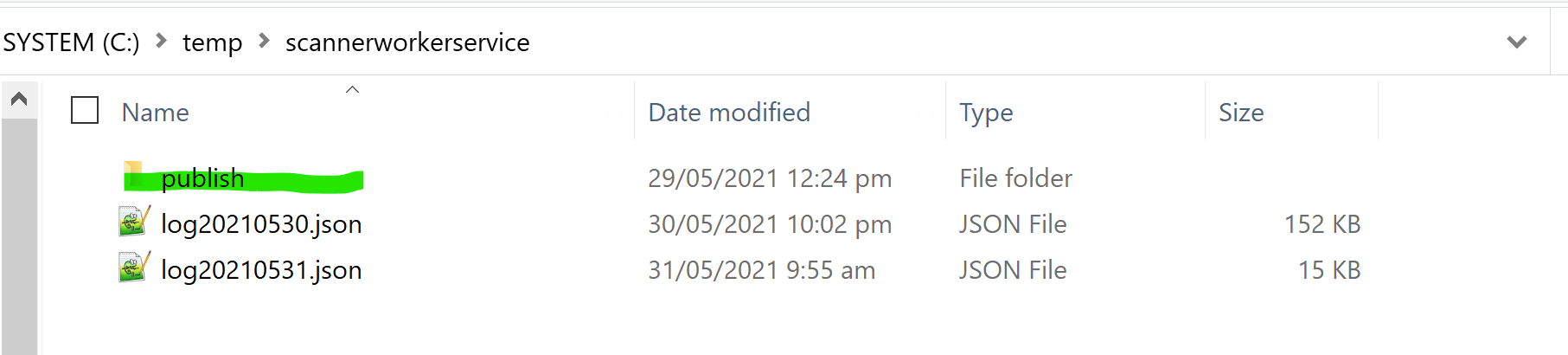
1. ScanEventApi already exists that will feed the worker application
2. Once the data fetched from the ScanEventApi will be stored either in a container based database or at the client-side such as IndexedDb or in a NoSQL approach, such as CosmosDB
3. To proof the concept and my comfort zone I have implemented with SQL Server for this exercise. Once it is verified and accepted I would be happy to move it where it is needed.
4. All the class objects defined under the applications are sample purpose only; which may differ in the real solution.
5. All the data supplied under the tables and tables itself
6. Also to limit or configure the worker application a sets of rules are under the table sy\_options. To test with different scenario please update the values to satisfy different situations. For example, limiting the application to fetch a number of events at a time by changing the value for limit\_events\_fetching from 100 to 20.

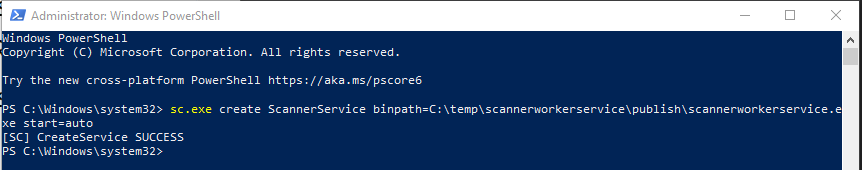
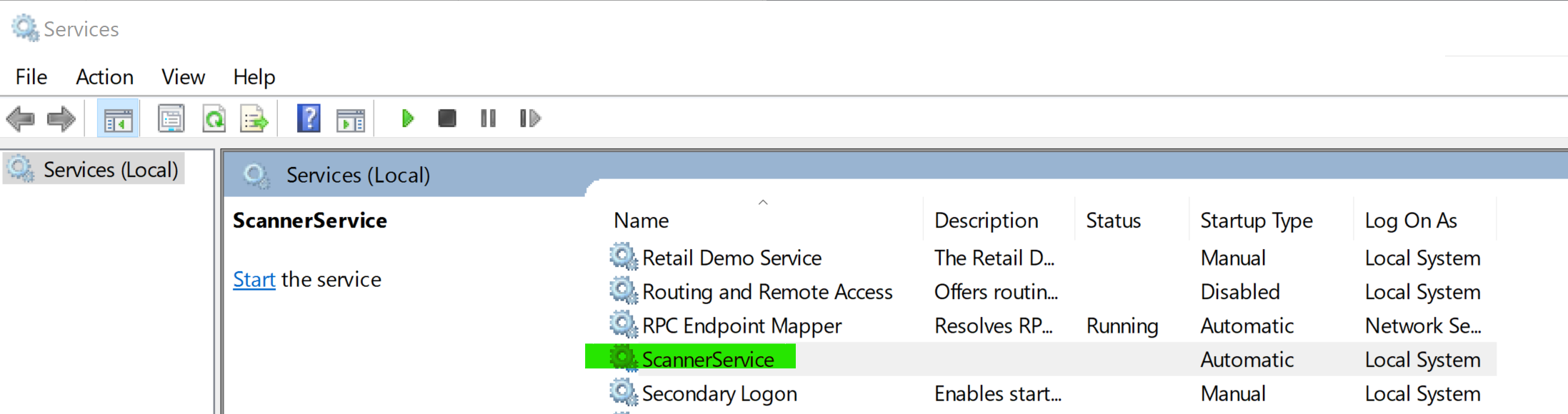
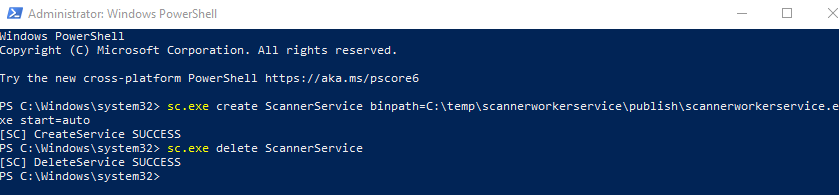
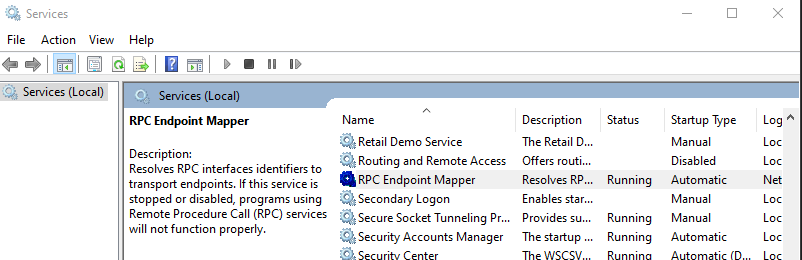
# Things to improve:

1. Need to secure the application for authentic access only
2. Need to catch any exceptions; which is very basic at the moment
3. Need to have through testing
4. Where to summarise the logged data/scanned data that are storing in the table ScanRecords and the log file logYYYYMMDD.json, sample attached

# Deploying ScannerWorkerService under the Windows services

* Before trying deployment we need to publish the ScannerWorkerService application.

For my environment I have published here to easy access

* Once the publish finished successfully
* Open Windows Powershell as Administrator
* And execute the following to register the service
  + e.g. sc.exe create ScannerService binpath=C:\temp\scannerworkerservice\publish\scannerworkerservice.exe start=auto
  + 
  + 
* If needs to unregister the service, execute the following (Please stop the service before unregister)
  + e.g. sc.exe delete ScannerService
  + 
  + 

# Conclusion:

In case if you have any questions or need clarification, please do not hesitate to reach out me.