Truck Monitor.

Assignment from Scania AB
Developed by Muhammad Ali Nazar
27/11/2017

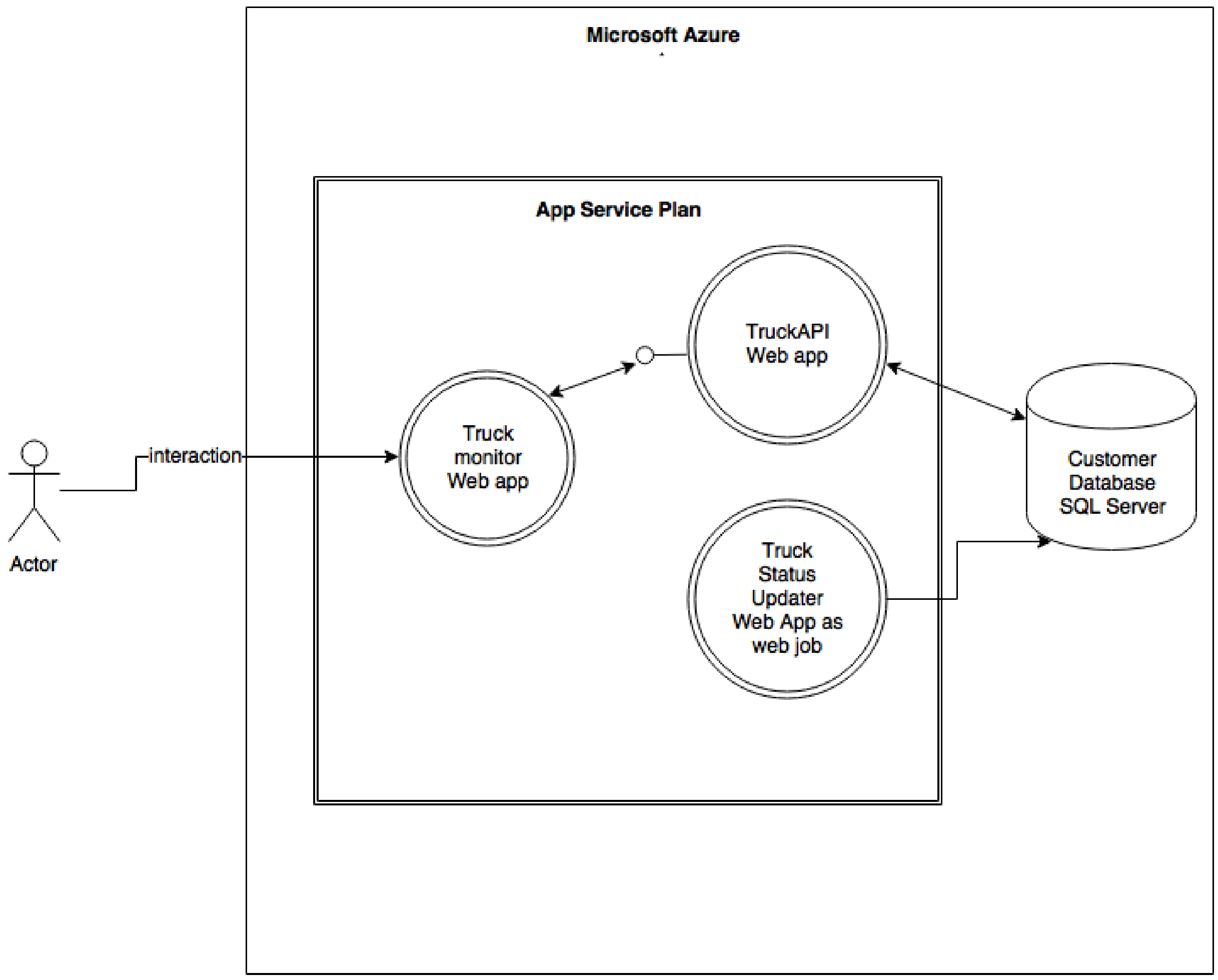
Introduction

A live demo of the solution is available on the following link:
http://truckmonitor.azurewebsites.net/

The data will load into the table after 10 seconds. Then the “Truck Connection Status” will be updated for the vehicles. Note that after the first initial data load into table, it can take upto 1 minute before you see any change in the “Truck Connection Status” in the table.

Solution

Architecture



Description of architecture

I have chosen to use Microsoft's cloud solution Azure to aid the development of the solution and to be able to give a live demo sample as provided in the introduction section above.
The solution is developed as a single page application using React and .net core 2.0 and consists of three applications and a test application:

1) TruckMonitor - The frontend React application that is deployed in a web app in Azure. This app visualizes the connection status of trucks in real time.

2) TruckAPI - The backend API developed in .net core 2.0 as a web api. The API is consumed by the Truck Monitor React application. The TruckAPI is deployed in a web app in Azure.
   The TruckAPI is secured from public access using JWT token authorization.

3) TruckStatusUpdater - This is a console application developed in .net core 2.0 that has the responsibility to read trucks from the database customerDatabase and simulate a ping request with a mocked response to each truck entry and then update each truck entry in the database with the response.
   This console application is deployed in a web app as a web job in Azure and is scheduled to run every minute.

The solution also includes a SQL database that exists in Azure in an SQL server that the TruckAPI and TruckStatusUpdater applications connects to and consumes.

Using the small applications above we get a micro service architecture as asked in the assignment requirements.

Elaboration of solution

I believe this architecture is well suited for the solution because it enables the opportunity to breakdown a big solution into smaller independent modules that can be separately developed and maintain. And by creating a universal API like TruckAPI, the modules can easily read and write data through one application.
Each application in the solution can be independently developed, deployed and maintained.
Microsoft Azure enables us to easily and effectively create application infrastructure with effective deployment strategies that speeds up development.