Azure Manual deployment guide

Tuesday, July 23, 2019 2:01 PM

===== Setting up Azure Resources

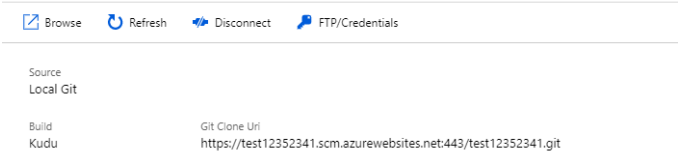
1. Log into your Azure Account at portal.azure.com.
2. Create a resource group and add the following resources to the new resource group.

Azure Resources	Used by	Description
App insights	GVA, Visibility Portal, GW listener	For logging across the system
IoT Hub	Gateways, GVA	Data, device management of Field GWs
Virtual Machine	GVA – Shipping API, GW listener, GW messenger, KeyStore	Hosts the GVA docker
Web APP	Visibility Portal	VP can also be hosted on a VM, if required..
Function APP	GW listener	Dedicated listener to the iot hub and store incoming data to the cosmos db
SQL Server and Database	GVA, GW listener / messenger	Storage of shipment information
Cosmos DB	GVA -GW listener, Shipping api, KeyStore	Storing data coming from GW, keystore data
Blob Storage	GVA, Visibility portal, OBT	POD photos, Canned shipping address, OBT ref GVA urls, GW logs, etc
Google Location Services	Gw listener, VP	Optional – for Cellular triangulation, and displaying maps
Twillio SMS gateway	GW listener	Optional – for sending geofence alerts

===== GVA VM setup

1. Go to the VM page in Azure, in the networks tab open the VM ports for SSH, Shipping API, GW messenger & Keystore. Ie. 22, 3001, 3002, 3003.
2. Populate the env file that will be used by the GVA docker. "src/gva.env" . You will need to create a blob container for the gva photos in order to fully populate the env file.
3. Configure the VM using the guide "GVA Virtual Machine Setup.pdf"
4. Upon running the GVA docker, it will automatically configure the SQL DB.. Incase you see any issue, please refer to "GVA Internal Database Setup.pdf"… Pls note the internal db setup can be done prior to running the GVA docker.

===== VP setup

1. While creating the VP chose Runtime stack "Node 8.9", OS-Linux
2. Go to the web app page on the azure portal.
3. First setup the environment variables for the VP, this can be done in the Configuration Tab, Application Settings section.
4. Description of variable that need to be setup are mentioned in the file - "VP env variable file template.env". These variables will need to be setup based on the resource created above.. For 'GVAurl' please add the url for the GVA VM created above.. This will typically be 'http' unless you have enabled SSL certificated in the GVA.
5. Got to the deployment center tab.. Here we will setup deployment via "local GIT". You can choose any other method as you choose.
6. In the "Build Provider" tab, choose KUDU "App Service build service",, then click finish..
7. Now a GIT uri will get generated to push the code:
8. 

9. Copy the uri, and also setup the credentials in the "FTP/Credentials" tab..
10. Git clone this url on your local pc.. Your should see an empty project.
11. Then copy the VP source code into the directory. Commit it. The push to origin.. After the push is done, the VP should start working

===== Setting up Function App for GW listener

1. Unless explicitly uncommented, the GW listener running in the GVA VM will not process sensordata messages coming from the GW. For this, we will rely on the function app to copy the sensordata to the cosmos DB.
2. Go to the "Platform Features" senction of the function app..
3. In the Configuration Section, add the following Application Settings
4. GOOGLE_API_KEY - for cellular triangulation, MONGODB_PASSWORD - password for the cosmosdb, MONGODB_USER - username for the cosmo db, MONGODB_URL - url for the cosmodb
5. Now add a new function; Click on the '+' icon
6. Choose IoT Hub (Event Hub).
7. Provide a function name, Event hub connection string corresponding to the above IOT HUB, and a new consumer group (create on in the iot hub, i.e. 'listener')
8. Once the function is created, go to the 'Integrate' section and setup trigger properly.. Provide the 'Event Hub Name', 'Consumer Group', confirm if the correctly reflect the ones in the IOT HUB.. Also make the 'Cardinality' as 'One'
9. In the function app file editor, create a copy all the file from: 'src/gwlistener/azure-function-app/iothub_gwlistener'.. This include 'index.js', gelocation.js, package.json, etc. avoid overwriting the function.json file..
10. Once the files have been created in the function app.. The function app can be restarted..
11. Upon restart a node_modules folder should get created and all the npm packages should get installed. If not, go to 'Platform Features' -> Console, and run "npm install" in the apropriate directory to manually create the node_modules.