Hands-on

Hands-on lab – Remote monitoring

LAB 1: Deploy PCS Remote Monitoring (Preview) and review the code.

LAB 2: Add a "physical" device to the solution

LAB 3: Add Time Series Insights to the solution

LAB 4: Working with device state using device twins

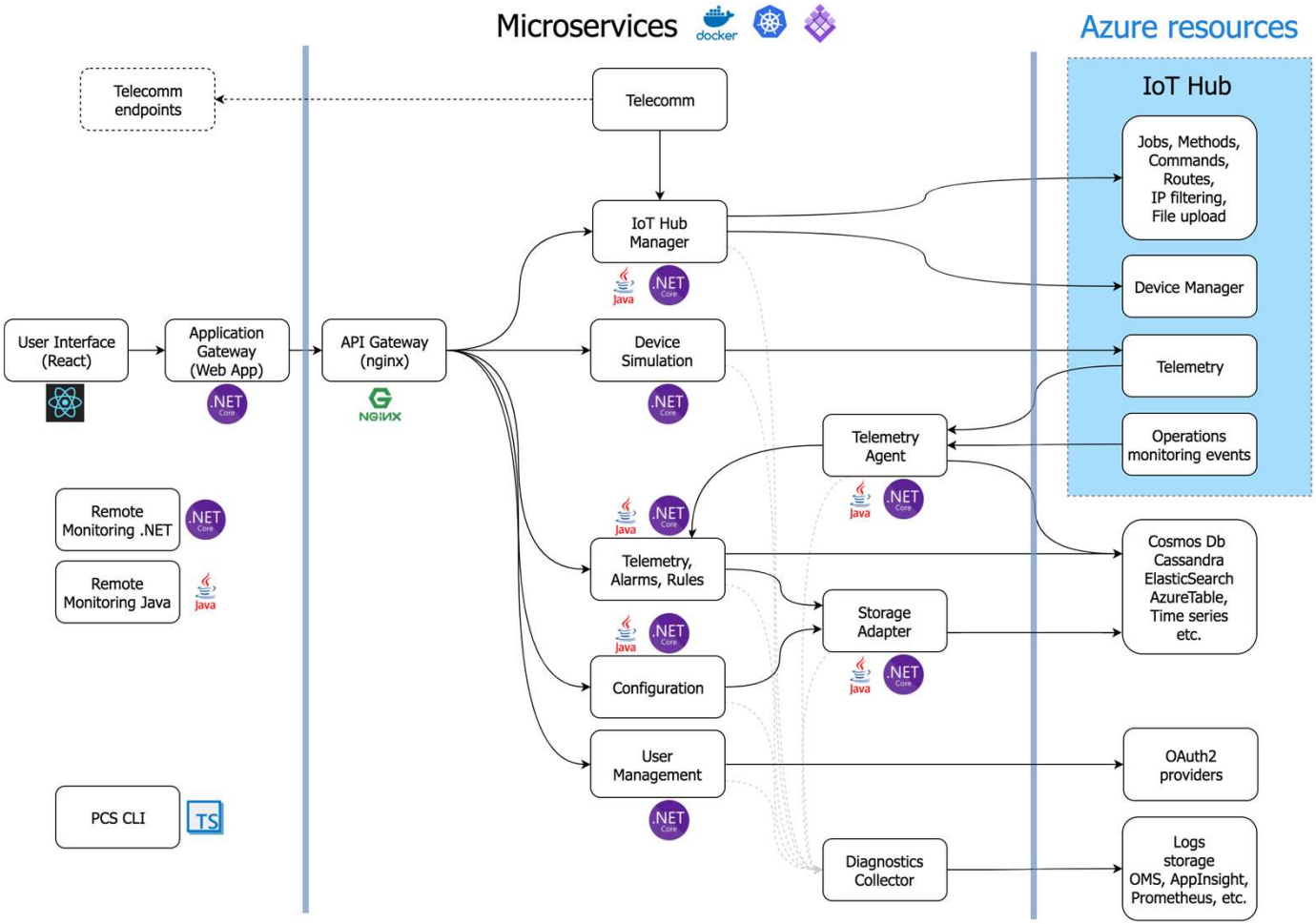
1) Set up desired properties

2) Change "physical" device to act upon change

LAB 5: Create a IoT Hub Job

LAB 6: Add a logic app for Enterprise Integration

PCS Remote Monitoring (preview)



LAB 1: Deploy Remote Monitoring (Basic)

Deploy the preconfigured solution

Before you deploy the preconfigured solution to your Azure subscription, you must choose some configuration options:+

1. Log on to azureiotsuite.com using your Azure account credentials, and click + to create a solution.
2. Click **Select** on the **Remote monitoring** tile.
3. On the **Create Remote Monitoring solution** page, enter a **Solution name** for your remote monitoring preconfigured solution.
4. Select a **Basic** or **Standard** deployment. If you are deplying the solution to learn how it works or to run a demonstration, choose the **Basic** option to minimize costs.
5. Choose either **Java** or **.NET** as the language. All the microservices are available as either Java or .NET implementations.
6. Review the **Solution details** panel for more information about your configuration choices.
7. Select the **Subscription** and **Region** you want to use to provision the solution.
8. Click **Create Solution** to begin the provisioning process. This process typically takes several minutes to run.
9. Now that you have deployed the remote monitoring solution, explore the capabilities of the solution dashboard

Solution types

Remote monitoring

Connect and monitor your devices to analyze untapped data and improve business outcomes by automating processes.

Select

Remote monitoring

Connect and monitor your devices to analyze untapped data and improve business outcomes by automating processes.

Preview highlights include:

- Redesigned user interface
- Microservices-based architecture
- Availability in both .NET and Java
- View an interactive demo

Select

Connected factory

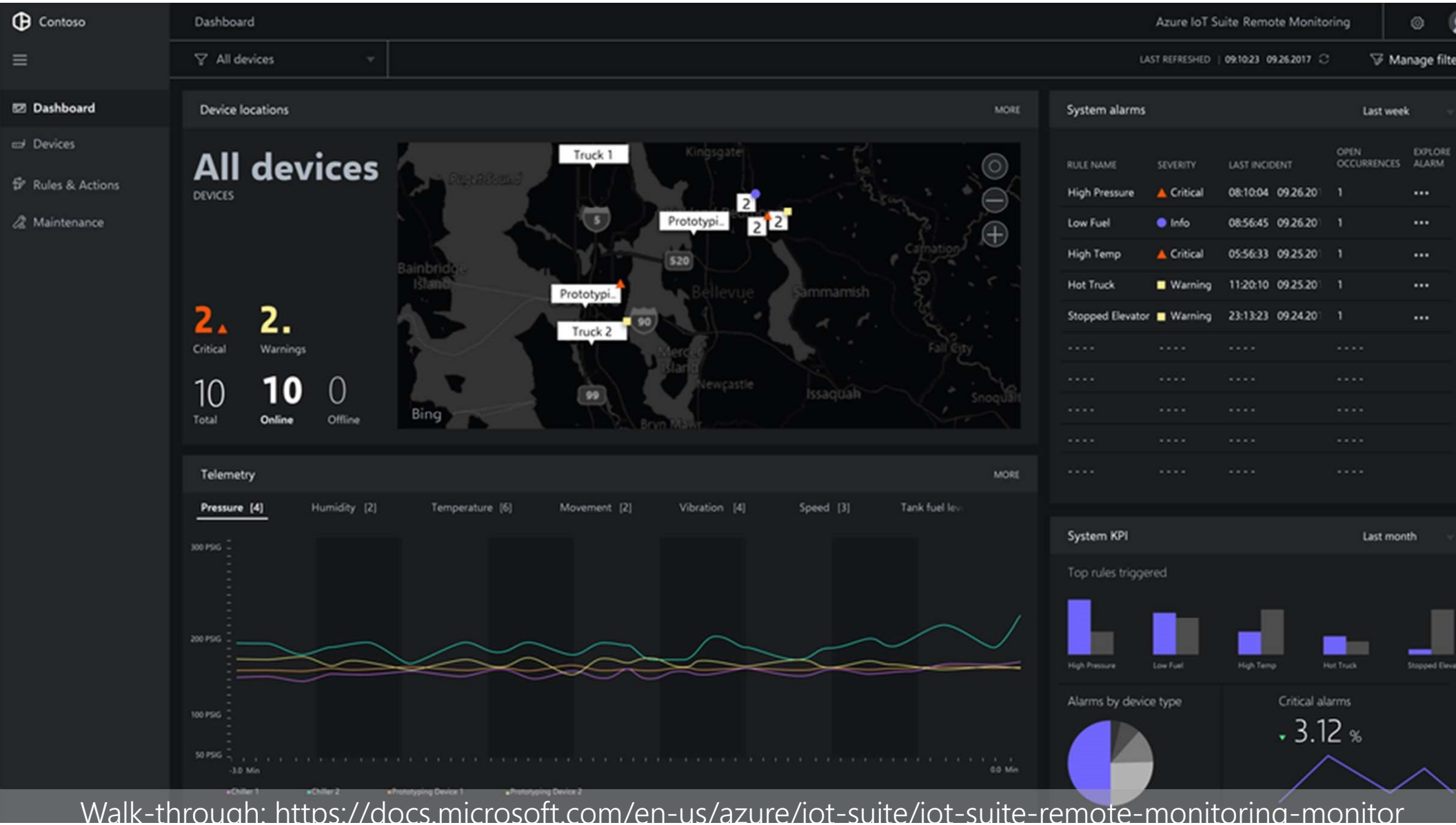
Accelerate your journey to Industrie 4.0 – connect, monitor and control industrial devices for insights using OPC UA to drive operational productivity and profitability.

Select

Predictive maintenance

Anticipate maintenance needs and avoid unscheduled downtime by connecting and monitoring your devices for predictive maintenance.

Select



Walk-through: https://docs.microsoft.com/en-us/azure/iot-suite/iot-suite-remote-monitoring-monitor

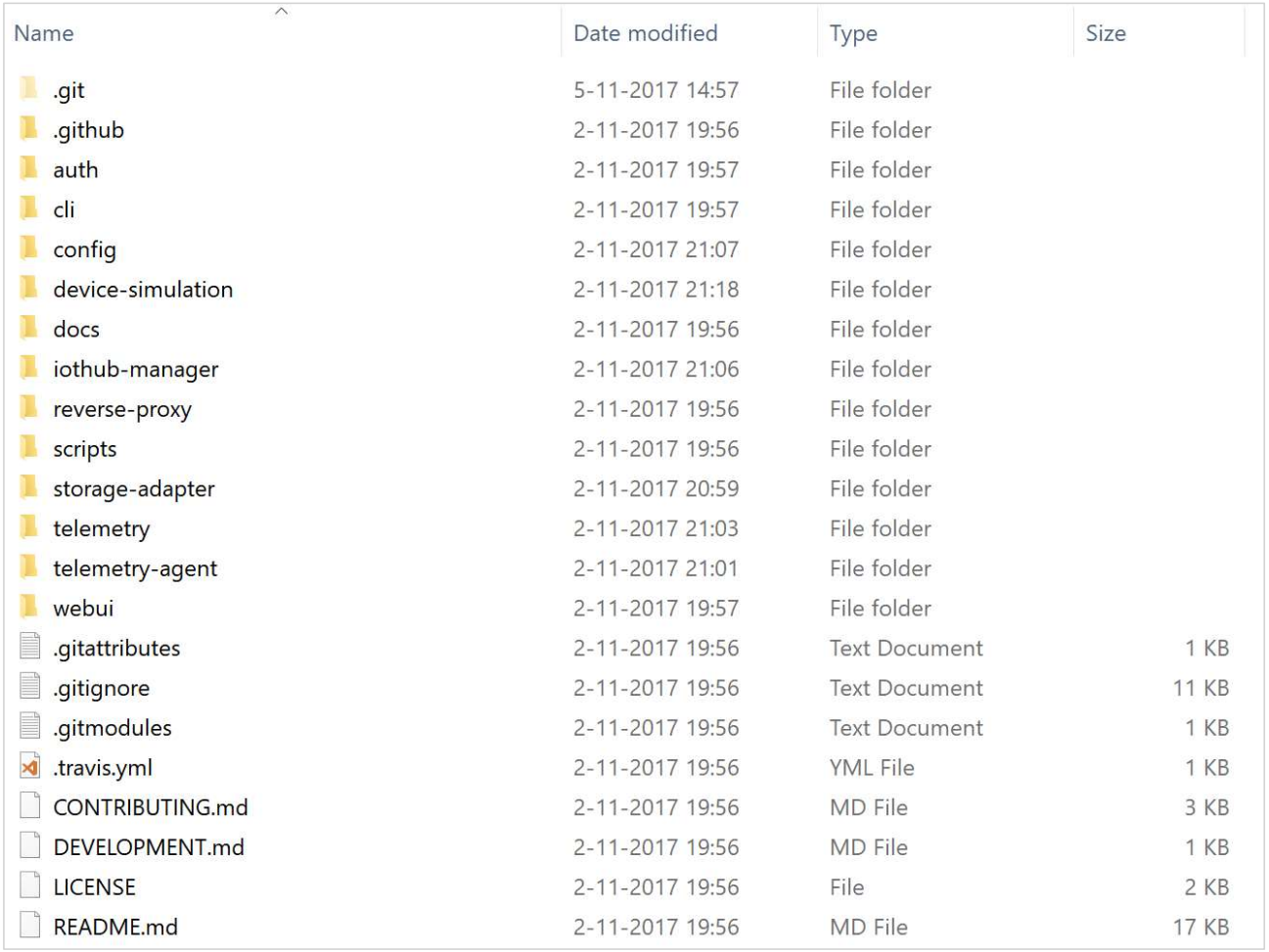
LAB 1: Get the code

Setup

1. Install git for Windows: https://git-scm.com/download/win
2. Open a terminal window or command shell
3. Create a directory to hold the code
4. Check out the repository and submodules

```
git clone --recursive https://github.com/Azure/azure-iot-pcs-remote-monitoring-dotnet.git
```

LAB 1: Code walk-through



Name	Date modified	Type	Size
.git	5-11-2017 14:57	File folder	
.github	2-11-2017 19:56	File folder	
auth	2-11-2017 19:57	File folder	
cli	2-11-2017 19:57	File folder	
config	2-11-2017 21:07	File folder	
device-simulation	2-11-2017 21:18	File folder	
docs	2-11-2017 19:56	File folder	
iothub-manager	2-11-2017 21:06	File folder	
reverse-proxy	2-11-2017 19:56	File folder	
scripts	2-11-2017 19:56	File folder	
storage-adapter	2-11-2017 20:59	File folder	
telemetry	2-11-2017 21:03	File folder	
telemetry-agent	2-11-2017 21:01	File folder	
webui	2-11-2017 19:57	File folder	
.gitattributes	2-11-2017 19:56	Text Document	1 KB
.gitignore	2-11-2017 19:56	Text Document	11 KB
.gitmodules	2-11-2017 19:56	Text Document	1 KB
.travis.yml	2-11-2017 19:56	YML File	1 KB
CONTRIBUTING.md	2-11-2017 19:56	MD File	3 KB
DEVELOPMENT.md	2-11-2017 19:56	MD File	1 KB
LICENSE	2-11-2017 19:56	File	2 KB
README.md	2-11-2017 19:56	MD File	17 KB

Each of the modules has a Visual Studio solution. Open a solution and review the code.

Understand how to customize the solution:

https://docs.microsoft.com/en-us/azure/iot-suite/iot-suite-remote-monitoring-customize

LAB 2: Adding a “physical” device to the solution

In this LAB, you implement a two devices that sends telemetry to the remote monitoring preconfigured solution using node.js. One chiller and one custom device

1. Create a new device in the Remote Monitoring solution.
2. Open bash on Windows 10 (activate if needed).
3. Create a Chiller Node.js solution using the online tutorial https://docs.microsoft.com/en-us/azure/iot-suite/iot-suite-connecting-devices-node#create-a-nodejs-solution.
4. Start the node.js solution and see the outcome on the Remote Monitoring Solution.

LAB 2: Adding a "physical" device to the solution

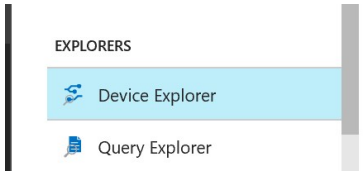
1. Create a new device on the Remote Monitoring solution.
2. Copy the *remote_monitoring.js* file to *my_custom_device.js*
3. Define completely new telemetry data for the device and adjust the .js file to represent them. Don't forget to adjust the *connectionString* to the new device.

LAB 3: Add Time Series Insights to the solution

1. Create a new Time Series Insights environment in the Azure portal
   https://docs.microsoft.com/en-us/azure/time-series-insights/time-series-insights-get-started
2. Create the IoT Hub event source for your Time Series Insights environment using the Azure portal
   https://docs.microsoft.com/en-us/azure/time-series-insights/time-series-insights-how-to-add-an-event-source-iothub
3. Grant data access to a Time Series Insights environment using Azure portal
   https://docs.microsoft.com/en-us/azure/time-series-insights/time-series-insights-data-access
4. Access your Time Series Insight environment
   https://insights.timeseries.azure.com/
5. Have a look at the Temperature by Device Id. Add pressure to the time series.

LAB 4: Working with device state using device twins

1. Open the Azure Portal and browse to your IoT Hub
2. Select your “physical” device in the Device Explorer



3. Open the device twin

4. Add latitude & longitude to the desired properties

```
6     "desired": {
7       "Latitude": 40.343432,
8       "Longitude": 8.3334,
9       "$metadata": {
```

LAB 4: Working with device state using device twins

1. Open the node.js file of your device in bash
2. Add code in your device to react to changes of the device twin

```
twin.on('properties.desired', function (delta) {
  // Handle desired properties set by solution
  console.log('Received new desired properties:');
  console.log(JSON.stringify(delta));
  // update location
  reportedProperties.Latitude = delta.Latitude;
  reportedProperties.Longitude = delta.Longitude;
  // Send updated properties
  twin.properties.reported.update(reportedProperties, function (err) {
    if (err) throw err;
    console.log('twin state reported');
  });
});
```

3. Start your device and change the values in the device twin
4. Open the Remote Monitoring Solution to see the change

LAB 5: Create a IoT Hub job

1. Adjust your "physical" device and add a direct method "ReactOnJob"
   a) Add "ReactOnJob" to SupportedMethods
   b) Add the client.onDeviceMethod for "ReactOnJob"
   c) Implement the defined method to update a property.
2. Create a nodejs job "jobService"
   a) Adjust the steps in https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-node-node-schedule-jobs#schedule-jobs-for-calling-a-direct-method-and-updating-a-device-twins-properties to represent your device and direct method.
3. Run the physical device and device job on 2 separate bash windows to see the output.

Use https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-node-node-schedule-jobs as your guideline

LAB 6: Add a logic app for Enterprise Integration

1. Open the Azure Portal and add a Logic App using the following walk-through: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-monitoring-notifications-with-azure-logic-apps
   a) Create a service bus
   b) Add an endpoint and routing rule
      Query string: temperature > ??. (whatever you want to use as trigger)
   c) Create and configure the Logic App
   d) Test it adjusting your "physical" device

```
function generateRandomIncrement() {
    return ((Math.random() * 5) - 1);
}
```