



DEVELOPING MULTITENANT APPLICATIONS ON SAP CLOUD PLATFORM

CPL282

Exercises / Solutions

Anuj Mehta / SAP

Mahipal Ramachandran / SAP

Ashitha MS / SAP

TABLE OF CONTENTS

Introduction.....	3
Part 1: Persistence Service multitenancy – Data initialization and isolation	6
Step 1: Review that data isolation using a Tenant Discriminator Column has been setup.....	8
Step 2: Review that the data initialization servlet has been setup and exposed.....	11
Step 3: Use the data initialization servlet to initialize pollution data for consumer ABC PetroCorp	13
Part 2: Identity Management – Assign application roles to users in SAP CP	17
Step1: Application logic for Plant Management based upon different roles	17
Step2: Roles assignment in SAP Cloud Platform cockpit	19
Step3: Validation of Identity Management setup.....	21
Part 3: On-Premise Connectivity service – On premise setup and configure connectivity service	23
Step 1: Expose on-premise services securely using SAP Cloud Connector	24
Section 1: Simulate on-premise setup by running a mock service on your machine.....	24
Section 2: Expose the mock service using SAP Cloud Connector	30
Step 2: Configure connectivity between exposed on-premise services and the cloud application using SAP CP destinations.....	36
Section 1: Configure Destinations from SAP CP.....	36
Section 2: Access the on-premise service from the cloud Java application	39
Part 4: Configure a Connectivity service and Test the multitenant application.....	43
Step 1: Configure connectivity on the consumer account (to service running on SAP CP).....	43
Step 2. Test the Pollution Monitoring dashboard by login to HTML5 Application	49
Part 5: Fiori Launchpad – Configuring tile for multi-tenant application	50
Step1: Login to SAP CP Portal Service and Import SAP Fiori Launchpad content into ABC PetroCorp Subaccount	50
Step2: Publish and Launch SAP Fiori Launchpad site from ABC PetroCorp subaccount	55

INTRODUCTION

'ITeLO Consulting' is a trusted SAP partner company that specializes in providing solutions for the **Oil and Gas industry**. They have adopted the **SAP Cloud Platform** as the innovation platform to build and run their applications on the cloud. They choose to deliver their solutions via the **Software as a service (SaaS) model** in which the software is deployed as a hosted service and is available to customers over the internet.

ITeLO Consulting has developed a **pollution monitoring dashboard** that helps Oil & Gas companies view a comparison of the air pollution levels onsite (at the plant) vs. the air pollution levels at the city (where the plant is located). They have found interest for this solution from certain Oil & Gas companies that want to adopt measures for proactive pollution prevention.

'ABC PetroCorp' and **'XYZ EnergyCorp'** are two such Oil & Gas companies based out of Canada and US respectively, that are buying this solution from ITeLO.

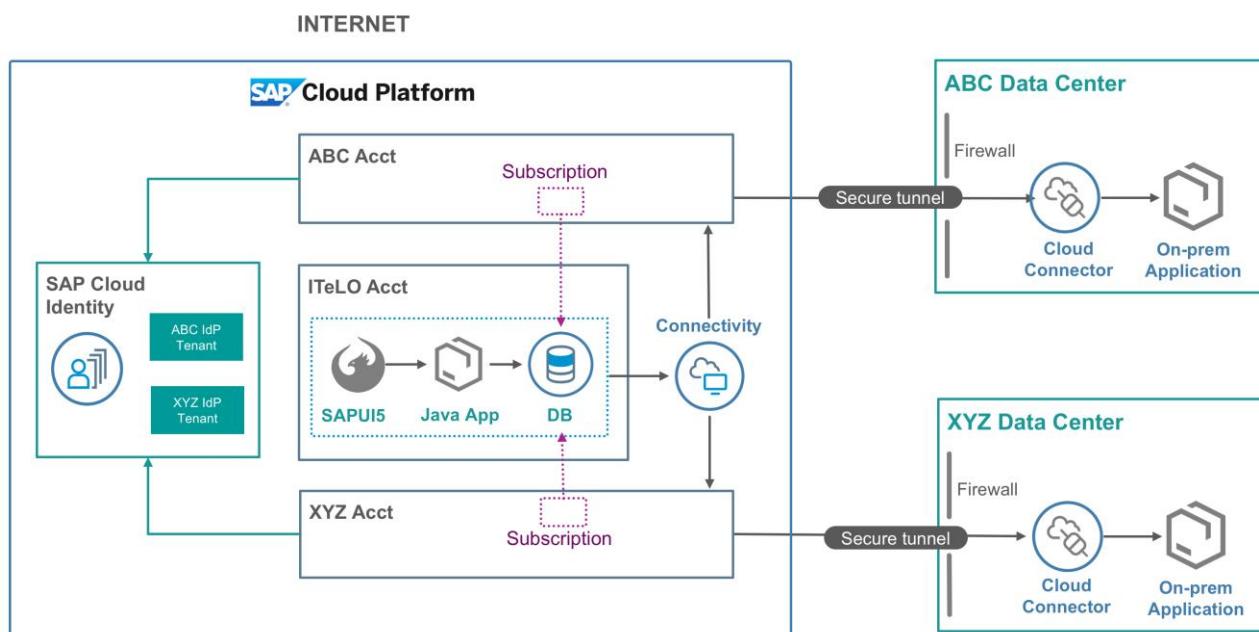
'Robert' works for ITeLO Consulting and is the architect for this project. ITeLO has bought an SAP Cloud Platform package enabled with the services required to build this application.

Robert's scenario is to develop a true **multi-tenant SaaS application** and deliver it to his customers.

'Emily' is an employee of the IT department of ABC PetroCorp. She has been assigned to work on this project with the Robert.

For XYZ EnergyCorp, **'Stephen'** is an employee of the IT department

The architecture of this multitenant application looks like this:



ITeLO Consulting (Provider)' subaccount

This is the provider account in which the database, business logic and user interface applications are running.

Database

The air pollution data of each plant is stored in the HANA DB using **SAP CP Persistence Service**. The data of different tenants are kept separate by using the tenant isolation features [provided by JPA](#).

Business Logic

The business logic is defined in the Java application which runs in the provider subaccount. The *same* instance of the Java application is run for different tenants. Each tenant creates a subscription to this Java application and gets a unique consumer access URL. This is the URL that will be used by end users to access the application. Whenever the application is accessed using this URL, SAP CP can [extract the tenant ID](#) that can be used to identify the tenant in the application logic.

User Interface

The UI logic is defined in an HTML5 application that runs on the provider account. Just like the Java application, the same instance of the HTML5 application is run for different tenants. And the application is accessed using a unique consumer access URL.

'ABC PetroCorp (Consumer)' subaccount

This is the consumer subaccount for ABC PetroCorp in which subscriptions are created and tenant specific configurations – such as the Identity Provider and connectivity to on-premise services – are made.

Remember that the Java/HTML5 applications are *not* deployed or copied to this account.

Identity and Access Management

Only employees of ABC PetroCorp should be allowed to access the subscribed applications.

ABC PetroCorp has its own Identity Provider (SAP HANA Cloud Identity Authentication service) that contains information about their employees such as name, email id, organization, role etc. This Identity Provider will be configured in the *ABC PetroCorp (Consumer)* subaccount such that the subscribed applications can be accessed only by their employees.

Connectivity

ABC PetroCorp has on –premise services running within their landscape that provide information about their plants – details like no. of workers at each plant, area of the plant etc. All this information must be pulled from their landscape and provided to the subscribed application using the SAP Cloud Connector.

'XYZ EnergyCorp (Consumer)' subaccount

Consumer account for XYZ EnergyCorp. Similar to the ABC PetroCorp (Consumer) subaccount.

As outlined that ITeLO Consulting has developed a pollution monitoring SaaS application. To provide this as a multi-tenant SaaS to its customers, ITeLO must make use of the multi-tenant capabilities of SAP Cloud Platform. Let us see how this can be achieved. In the interest of saving your time, following tasks have already been done for you.

1. Project and sub-accounts set up and create subscriptions

Robert has created a subaccount for ITeLo Consulting and ABC PetroCorp.

Robert has deployed applications (Java, HTML5) build by development team of ITeLo Consulting in ITeLo Consulting (provider) subaccount

Robert has done the SAP HANA Database binding to the Java application in ITeLo Consulting (provider) subaccount.

Robert has subscribed ABC PetroCorp(consumer) subaccount to the Java and HTML5 applications running on the ITeLo Consulting (provider) subaccount.

2. Configure Open source service in the provider subaccount

As outlined above pollution monitoring dashboard application helps consumers to compare plant's pollution data with city's pollution data.

Here plant data comes from the plants but what's about city's pollution data?

The answer is [air quality service from OpenAQ](#) which is open source and publicly available over the internet. Robert has configured this service as a destination in the ITeLo Consulting (provider) subaccount.

3. Identity Management -Setup of Identity Authentication service

To provide security and identity management to Pollution monitoring dashboard application Emily has established trust between ABC PetroCorp(consumer) account and the ABC PetroCorp's IDP. She has also imported following end users required for the application in her ABC PetroCorp's IDP.

1. User:Johan ID: (Area manager)
2. User ID:Smith (PlantSupervisor)

Now we will focus on 'how Emily collaborate with Robert to do customer-specific configuration in the ABC PetroCorp (consumer) subaccount. This will be covered in the following parts:

Part1. Persistence service multitenancy – Data initialization and isolation

Part2. Identity Management – Assign application roles to users in SAP CP

Part3. On-Premise Connectivity service – On premise setup and configure connectivity service

Part4. Configure a Connectivity service and Test the multitenant application

Part5. Fiori Launchpad – Configuring tile for multi-tenant application

PART 1: PERSISTENCE SERVICE MULTITENANCY – DATA INITIALIZATION AND ISOLATION

Explore how to achieve persistence multitenancy on the SAP Cloud Platform.

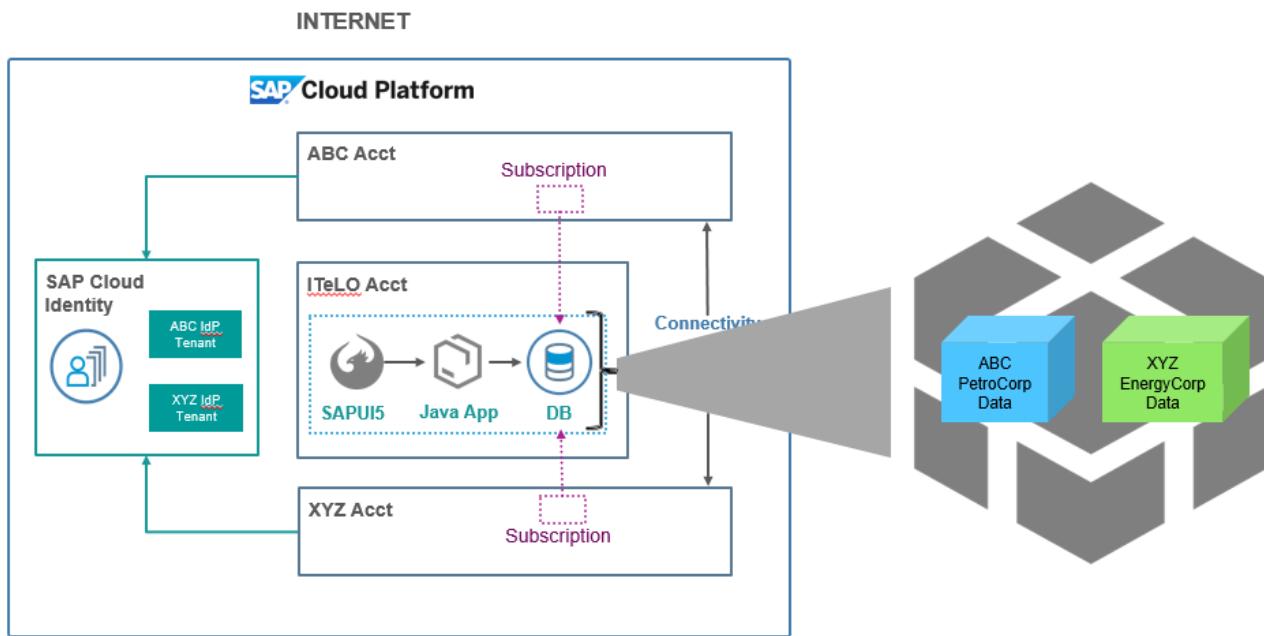
Initialize the Pollution data required for the application using an initialization servlet.

Time for completion: 10min

'ITeLO Consulting' is a trusted SAP partner company that specializes in providing solutions for the **Oil and Gas industry**. They have adopted the **SAP Cloud Platform** as the innovation platform to build and run their applications on the cloud.

As outlined earlier, ABC PetroCorp and XYZ EnergyCorp are Oil and Gas companies who are interested in purchasing ITeLO Consulting's innovative **Pollution Monitoring** solution on the SAP Cloud Platform. As the foundation for a true multitenant application, ITeLO Consulting should ensure that the data of ABC PetroCorp is completely isolated from the data of XYZ EnergyCorp in the database.

Robert from ITeLO Consulting wants his application to isolate the data storage of his customers, ABC PetroCorp and XYZ EnergyCorp. This is very important to ensure the security of customer's data and the application. Robert is wondering how this can be achieved?



To achieve this data isolation, we need to decide the granularity of isolation i.e. Table-level (store customer data in different tables in a single schema), Schema-level (store customer data in different schema), Tenant-level (store customer data in different database tenants). The SAP Cloud Platform Persistence Service, in conjunction with JPA (Java Persistence API) and EclipseLink, supports the implementation of these data-isolation methods.

There are three possible methods to achieve database/persistence multitenancy.

Using Tenant Discriminator Column: Uses a column in the table to ensure data isolation. In this approach a single DB schema is shared between all application consumers. The tenant identifier provided by the Tenant Runtime can be used as the value in the discriminator column.

Code Snippet:

```
@Entity  
@Table(name = "PERSON")  
@Multitenant  
@TenantDiscriminatorColumn(name = "TenantId", contextProperty = "eclipselink-tenant.id", length = 36)  
@NamedQuery(name = "AllPersons", query = "select p from Person p")  
public class Person {  
    ...  
}
```

Using Table-per-tenant multitenancy: Uses tenant-specific tables to ensure data isolation. In this approach a new table or schema in the database is created for each customer. A tenant table discriminator specifies how to discriminate one customer's table from the other. The tables can be in the same schema, using a prefix or suffix naming pattern to distinguish them; or they can be in separate schema, using a schema tenant table discriminator.

Code Snippet:

```
@Entity  
@Table(name = "EMPLOYEE")  
@Multitenant(TABLE_PER_TENANT)  
@TenantTableDiscriminator(type=SCHEMA, contextProperty="eclipselink-tenant.id")  
public class Employee {  
    ...  
}
```

Using Multitenant Database Containers: Uses different database containers/tenants for data isolation. In this approach a database new tenant is created for each customer.

In this lesson, let's focus on how to achieve data isolation using a Tenant Discriminator Column in the database. In addition, as a preparatory step, we will also initialize the data required for the Pollution Monitoring application being built as a part of the hands-on.

Robert will need to perform the following steps:

Step1: Review that data isolation using a Tenant Discriminator Column has been setup.

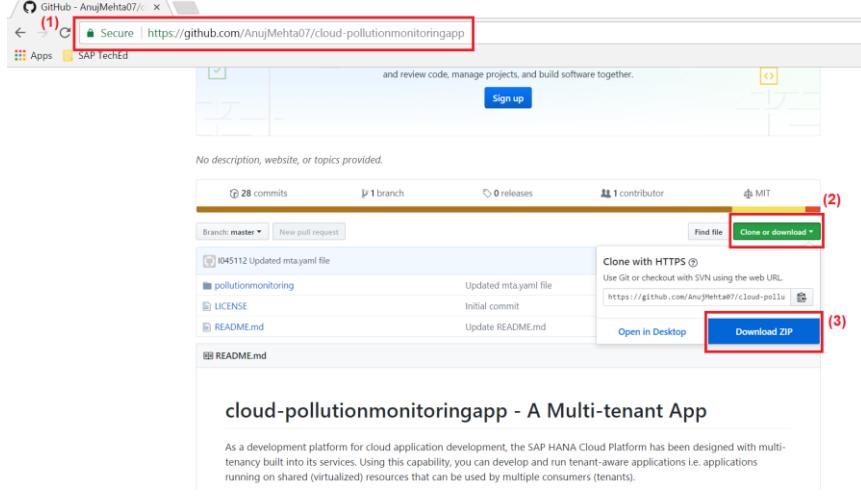
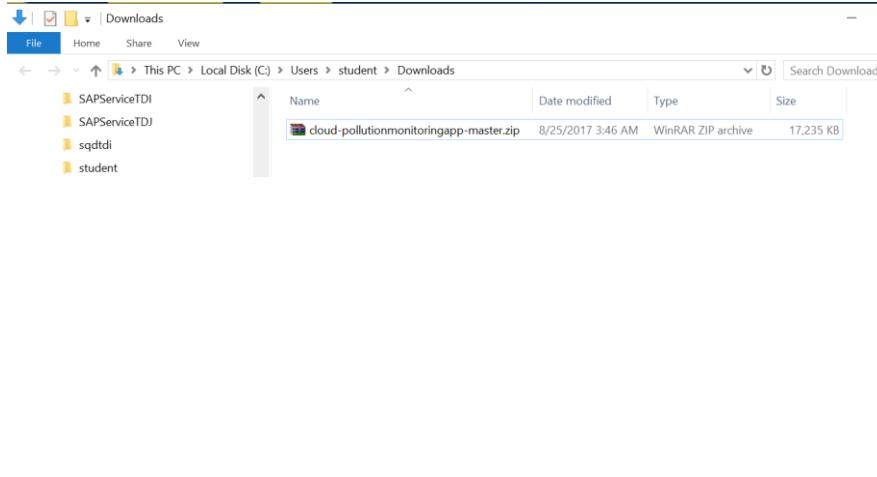
Step2: Review that the data initialization servlet has been setup so that customers can use it for their account.

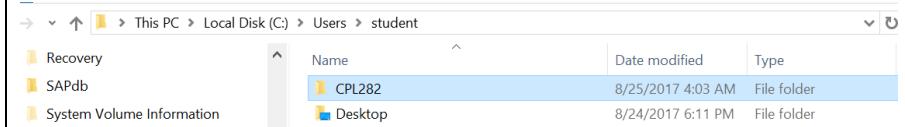
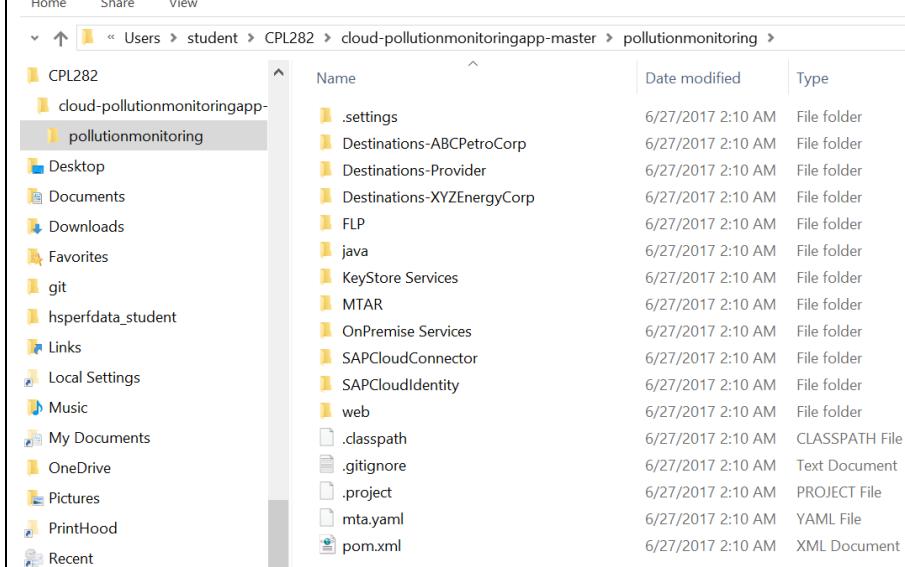
Emily will need to perform the following steps:

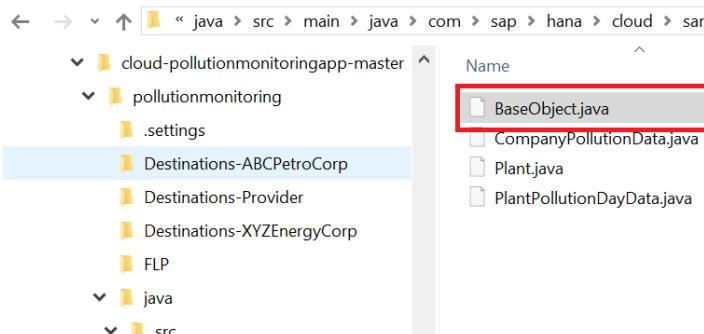
Step3: Use the data initialization servlet provided by Robert to initialize the data for ABC PetroCorp

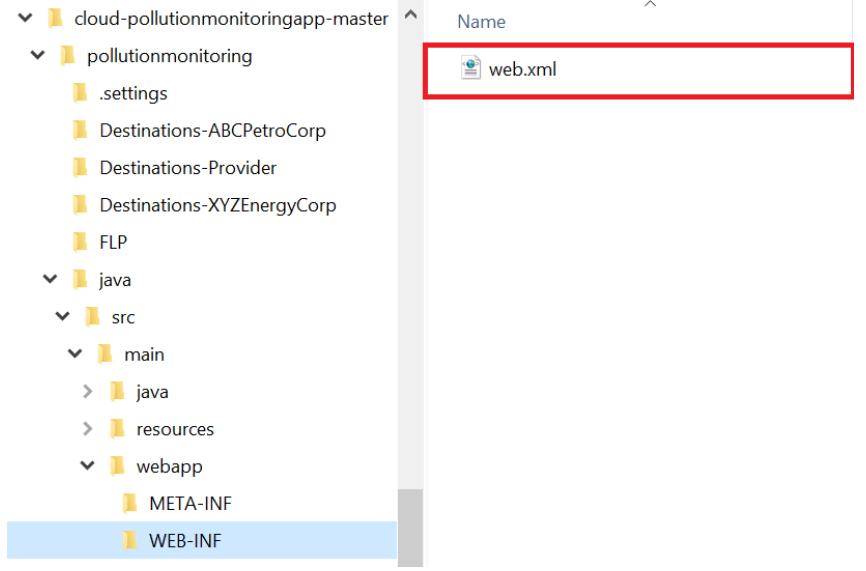
Step 1: Review that data isolation using a Tenant Discriminator Column has been setup.

In this step, you will import the pollution monitoring application to Eclipse and check how the data model of the application has been used to bring in persistence multitenancy.

Explanation	Screenshot
<p>1. Open the following URL https://github.com/AnujMehta07/cloud-pollutionmonitoringapp</p> <p>in the browser. Click on the Clone or download and Download ZIP to download the content of project pollutionmontroring.</p>	
<p>2. This will download the project into cloud-pollutionmonitoringapp-master.zip file in the Downloads folder (C:/Users/student/Downloads)</p>	

Explanation	Screenshot
<p>3.) Create a folder with name CPL282 in the location C:/Users/student</p>	 <p>This screenshot shows the Windows File Explorer interface. The path 'This PC > Local Disk (C) > Users > student' is selected. A new folder named 'CPL282' has been created and is highlighted in blue. Other folders visible include 'Recovery', 'SAPdb', and 'System Volume Information'. The file list on the right shows 'CPL282' was created on 8/25/2017 at 4:03 AM and is a 'File folder'. A 'Desktop' folder is also listed.</p>
<p>3.) Unzip the content of cloud-pollutionmonitoringapp-master.zip file under the CPL282 folder. The folder structure will look like this.</p>	 <p>This screenshot shows the Windows File Explorer interface with the path 'Users > student > CPL282 > cloud-pollutionmonitoringapp-master > pollutionmonitoring' selected. The 'pollutionmonitoring' folder is highlighted. The left sidebar shows standard Windows icons for desktop, documents, downloads, favorites, git, hiperdata_student, links, local settings, music, my documents, OneDrive, pictures, PrintHood, and recent files. The right pane lists the contents of the 'pollutionmonitoring' folder, which includes various subfolders and files such as '.settings', 'Destinations-ABCpetroCorp', 'Destinations-Provider', 'Destinations-XYZEnergyCorp', 'FLP', 'java', 'KeyStore Services', 'MTAR', 'OnPremise Services', 'SAPCloudConnector', 'SAPCloudIdentity', 'web', '.classpath', '.gitignore', '.project', 'mta.yaml', and 'pom.xml'. All files were created on 6/27/2017 at 2:10 AM.</p>

Explanation	Screenshot
<p>3. In the imported project, navigate to /pollutionmonitoring/java/src/main/java/com/sap/hana/cloud/samples/pollutionmonitoring/model/BaseObject.java. Open the java file with notepad++</p>	 <p>The screenshot shows a file explorer window with the following directory structure:</p> <ul style="list-style-type: none">cloud-pollutionmonitoringapp-masterpollutionmonitoring<ul style="list-style-type: none">.settingsDestinations-ABCPetroleumCorpDestinations-ProviderDestinations-XYZEnergyCorpFLPjava<ul style="list-style-type: none">src<ul style="list-style-type: none">main<ul style="list-style-type: none">java<ul style="list-style-type: none">com<ul style="list-style-type: none">sap<ul style="list-style-type: none">hana<ul style="list-style-type: none">cloudsamplespollutionmonitoringapimodel <p>The 'model' folder and the 'BaseObject.java' file within it are both highlighted with red boxes.</p>

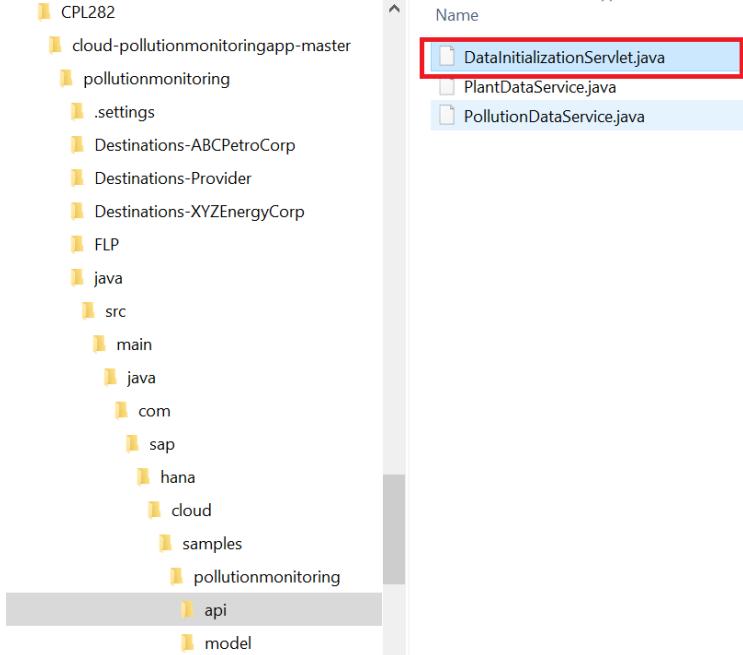
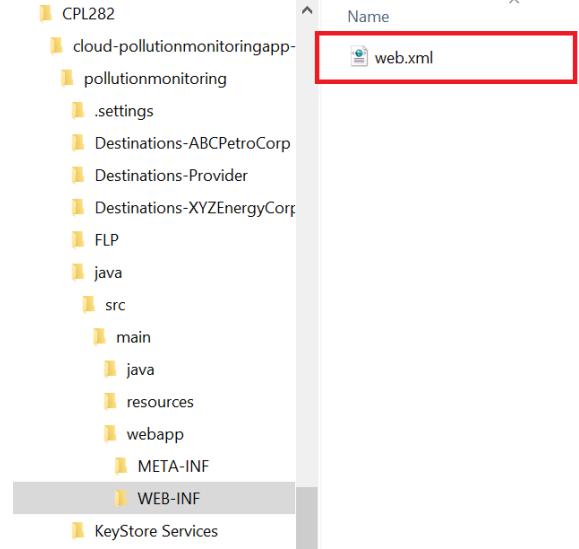
Explanation	Screenshot
4. Check how persistence multitenancy has been achieved using the Tenant Discriminator Column method.	<pre data-bbox="633 367 1481 530">@MappedSuperclass @Multitenant @TenantDiscriminatorColumn(name = "TENANT_ID", contextProperty="elipselink.tenant.id", length=1) public abstract class BaseObject { /** * ... */ }</pre>
5. Navigate to /pollutionmonitoring/java/src/main/webapp/WEB-INF/web.xml . Open the web.xml file with notepad++	 <p>The screenshot shows a file tree for the 'pollutionmonitoring' project. The structure includes 'cloud-pollutionmonitoringapp-master', 'pollutionmonitoring', '.settings', 'Destinations-ABCPetroCorp', 'Destinations-Provider', 'Destinations-XYZEnergyCorp', 'FLP', 'java', 'src', 'main', 'java', 'resources', 'webapp', 'META-INF', and 'WEB-INF'. A red box highlights the 'web.xml' file located in the 'WEB-INF' directory.</p>
6. Check that the data source (db) has been bound to the application as the default db.	<pre data-bbox="666 1170 1275 1305"><!-- welcome file list --> <resource-ref> <res-ref-name>jdbc/DefaultDB</res-ref-name> <res-type>javax.sql.DataSource</res-type> </resource-ref> <!-- resource-ref --></pre>

Result: We have now reviewed how multitenancy has been implemented for the Pollution Monitoring application.

Step 2: Review that the data initialization servlet has been setup and exposed.

Robert is aware that consumers of his application would require to load data for them to use it. So, he has built and exposed a data initialization servlet, which can be used by customers (like Emily) to push data into the Pollution Monitoring app.

In this step, you will check out how the data initialization servlet has been setup and how it has been exposed for consumption.

Explanation	Screenshot
<p>1. Check the servlet /pollutionmonitoring/java/src/main/java/com/sap/hana/cloud/samples/pollutionmonitoring/api/DataInitializationServlet.java</p>	
<p>2. Navigate to /pollutionmonitoring/java/src/main/webapp/WEB-INF/web.xml.</p>	
<p>3. Verify that the servlet is exposed using the url path '/initialize'</p>	<pre data-bbox="653 1669 1469 1838"><!-- --> <servlet> <servlet-name>DataInitializationServlet</servlet-name> <servlet-class>com.sap.hana.cloud.samples.mfplantapp.api.DataInitializationSe </servlet> <servlet-mapping> <servlet-name>DataInitializationServlet</servlet-name> <url-pattern>/initialize</url-pattern></pre>

Result: We have now reviewed how the data initialization servlet to load data into the Pollution Monitoring application has been setup and exposed. The setup on the Provider account is complete!

Step 3: Use the data initialization servlet to initialize pollution data for consumer ABC PetroCorp

Emily, from ABC PetroCorp will now access the Pollution Monitoring application using the subscription created (already setup for you).

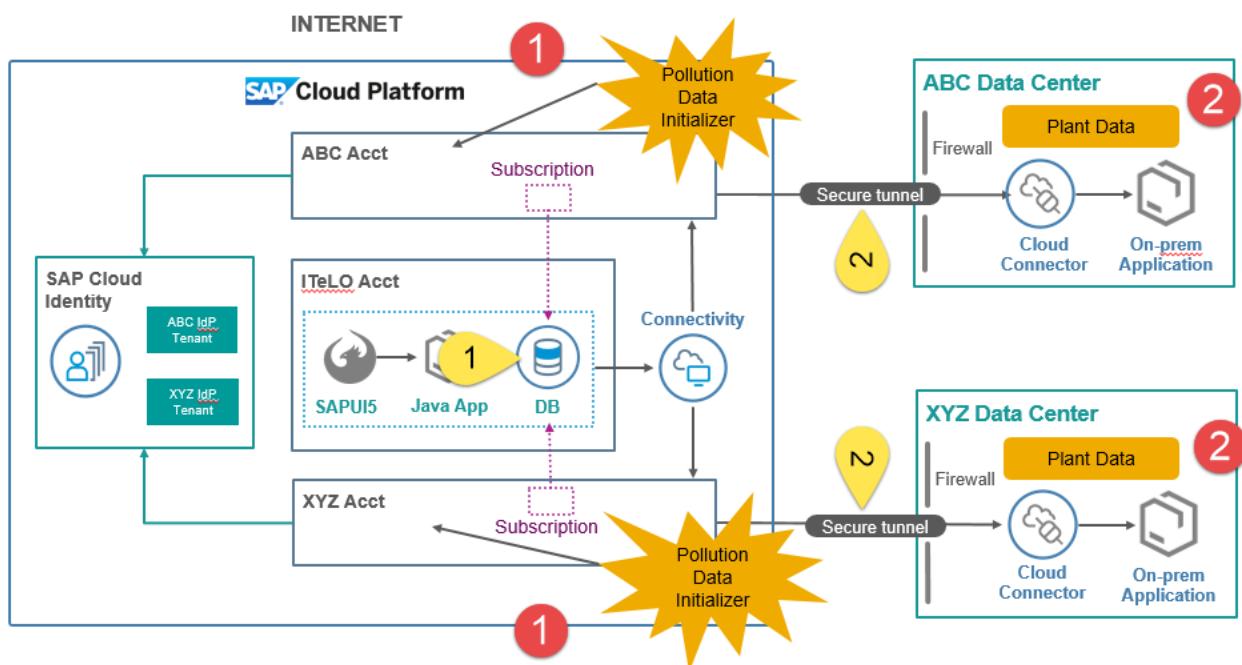
The Pollution Monitoring application, majorly deals with 2 types of data:

Pollution Data: which is read from sources like external APIs, Pollution sensors, etc.

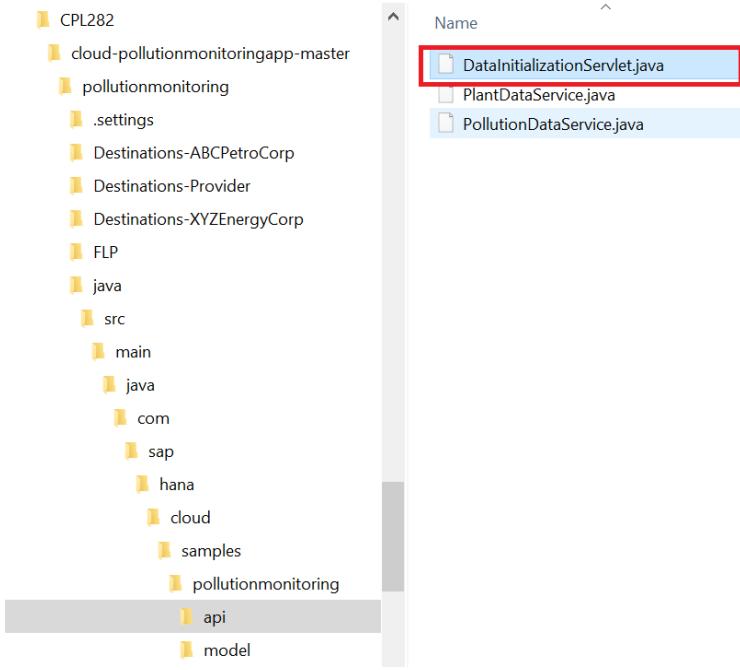
Plant Data: which is generally read from Systems of record like ERP systems, which are mostly On-Premise within the consumer private network.

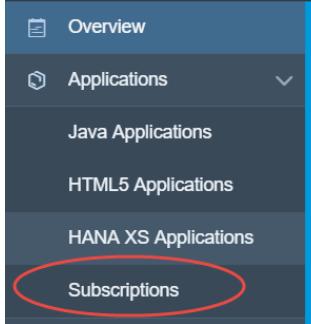
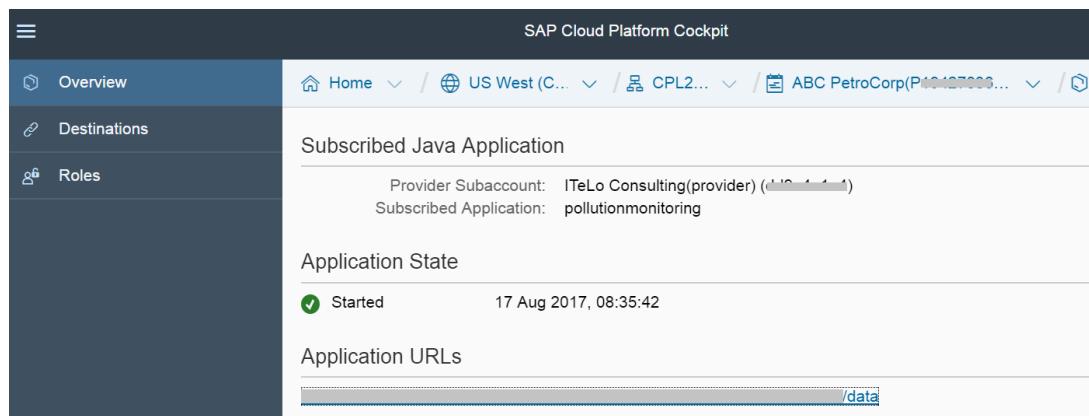
1) => In this step, you will create Pollution data using the Data Initialization servlet, which is included in the project. This data will be stored in the HANA DB on SAP CP, which is bound to the application in the provider account.

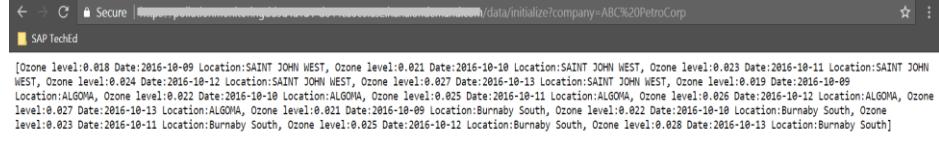
2) => The plant data is read 'on-demand' by the application, from the consumer system, i.e. it is not stored in the HANA DB on SAP CP. You will try this in Lesson 3 'On-Premise Connectivity service – On premise setup and configure connectivity service'.



In this step, you will load the Pollution Data into the subscribed application. To do this you will use the Data Initialization servlet in the project, which can be utilized to push simulated (prepared) data to the HANA database. Let us check how the prepared data looks like and how this can be initialized using the servlet.

Explanation	Screenshot
<p>1. Navigate to <code>/pollutionmonitoring/java/src/main/java/com/sap/hana/cloud/samples/pollutionmonitoring/api/DataInitializationServlet.java</code>. Open the java file.</p>	 <pre> CPL282 ├── cloud-pollutionmonitoringapp-master │ ├── pollutionmonitoring │ │ ├── .settings │ │ ├── Destinations-ABCPetroleumCorp │ │ ├── Destinations-Provider │ │ ├── Destinations-XYZEnergyCorp │ │ ├── FLP │ │ ├── java │ │ │ ├── src │ │ │ │ ├── main │ │ │ │ │ └── java │ │ │ │ │ └── com │ │ │ │ │ └── sap │ │ │ │ │ └── hana │ │ │ │ └── cloud │ │ │ └── samples │ │ └── pollutionmonitoring │ └── api └── model </pre>
<p>2. The prepared data for both the ABC PetroCorp and XYZ EnergyCorp can be seen in the code.</p>	<pre> if(parameter.equalsIgnoreCase(ABC_PETRO_CORP)) { fillPlantData("101","0.018","SAINT JOHN WEST","2016-10-09"); fillPlantData("101","0.021","SAINT JOHN WEST","2016-10-10"); fillPlantData("101","0.023","SAINT JOHN WEST","2016-10-11"); fillPlantData("101","0.024","SAINT JOHN WEST","2016-10-12"); fillPlantData("101","0.027","SAINT JOHN WEST","2016-10-13"); fillPlantData("102","0.018","ALGOMA","2016-10-09"); fillPlantData("102","0.022","ALGOMA","2016-10-10"); fillPlantData("102","0.025","ALGOMA","2016-10-11"); fillPlantData("102","0.026","ALGOMA","2016-10-12"); fillPlantData("102","0.027","ALGOMA","2016-10-13"); fillPlantData("103","0.021","Burnaby South","2016-10-09"); fillPlantData("103","0.022","Burnaby South","2016-10-10"); fillPlantData("103","0.023","Burnaby South","2016-10-11"); fillPlantData("103","0.025","Burnaby South","2016-10-12"); plantList = fillPlantData("103","0.028","Burnaby South","2016-10-13"); } else if(parameter.equalsIgnoreCase(XYZ_ENERGY_CORP)) { fillPlantData("201","0.027","Hampshire C64","2016-10-09"); fillPlantData("201","0.023","Hampshire C64","2016-10-10"); fillPlantData("201","0.021","Hampshire C64","2016-10-11"); fillPlantData("201","0.025","Hampshire C64","2016-10-12"); fillPlantData("201","0.019","Hampshire C64","2016-10-13"); fillPlantData("202","0.021","Garyville","2016-10-09"); fillPlantData("202","0.025","Garyville","2016-10-10"); fillPlantData("202","0.022","Garyville","2016-10-11"); fillPlantData("202","0.021","Garyville","2016-10-12"); fillPlantData("202","0.023","Garyville","2016-10-13"); fillPlantData("203","0.023","Arlington Municipal","2016-10-09"); fillPlantData("203","0.021","Arlington Municipal","2016-10-10"); fillPlantData("203","0.019","Arlington Municipal","2016-10-11"); fillPlantData("203","0.027","Arlington Municipal","2016-10-12"); plantList = fillPlantData("203","0.028","Arlington Municipal","2016-10-13"); } </pre>

Explanation	Screenshot
<p>You will initialize the pollution data for ABC PetroCorp by calling the Data Initialization servlet</p>	
<p>3. Browse the URL https://account.hana.ondemand.com/ to open SAP Cloud Platform cockpit, go to the ABC Petro Corp (P194278****) account.</p>	 <p>The screenshot shows the SAP Cloud Platform Cockpit interface for the ABC PetroCorp account. It displays the following information:</p> <ul style="list-style-type: none"> Java Applications: <ul style="list-style-type: none"> 0 Started 0 Failed 0 Stopped Java Quota: No Java Quota 4 Members: (with a pencil icon for editing)
<p>4. Navigate to Applications -> Subscriptions</p>	 <p>The screenshot shows the SAP Cloud Platform Cockpit navigation menu under the Applications section. The Subscriptions option is highlighted with a red oval.</p>
<p>5. In the Subscribed Java Application Overview page, copy the Application URLs.</p>	 <p>The screenshot shows the SAP Cloud Platform Cockpit Overview page for a subscribed Java application. Key details include:</p> <ul style="list-style-type: none"> SAP Cloud Platform Cockpit Home / US West (C... / CPL2... / ABC PetroCorp(P194278... / Subscribed Java Application Provider Subaccount: ITeLo Consulting(provider) (10-11-1) Subscribed Application: pollutionmonitoring Application State: Started (17 Aug 2017, 08:35:42) Application URLs: (with a 'data' link)

Explanation	Screenshot
<p>6. In a new browser window, paste the copied URL and add /Initialize?company=ABC PetroCorp to it.</p> <p>Initialization should complete and the data that was pushed to the application should be displayed.</p>	 <pre>[{"Ozone level":0.018,"Date":2016-10-09,"Location":SAINT JOHN WEST}, {"Ozone level":0.021,"Date":2016-10-10,"Location":SAINT JOHN WEST}, {"Ozone level":0.023,"Date":2016-10-11,"Location":SAINT JOHN WEST}, {"Ozone level":0.024,"Date":2016-10-12,"Location":SAINT JOHN WEST}, {"Ozone level":0.027,"Date":2016-10-13,"Location":SAINT JOHN WEST}, {"Ozone level":0.019,"Date":2016-10-09,"Location":ALGOA}, {"Ozone level":0.022,"Date":2016-10-10,"Location":ALGOA}, {"Ozone level":0.025,"Date":2016-10-11,"Location":ALGOA}, {"Ozone level":0.026,"Date":2016-10-12,"Location":ALGOA}, {"Ozone level":0.027,"Date":2016-10-13,"Location":ALGOA}, {"Ozone level":0.021,"Date":2016-10-09,"Location":Burnaby South}, {"Ozone level":0.022,"Date":2016-10-10,"Location":Burnaby South}, {"Ozone level":0.023,"Date":2016-10-11,"Location":Burnaby South}, {"Ozone level":0.025,"Date":2016-10-12,"Location":Burnaby South}, {"Ozone level":0.028,"Date":2016-10-13,"Location":Burnaby South}]</pre>

Result: We have now initialized the pollution data for ABC PetroCorp to ITeLO's Pollution Monitoring application.

Congratulations! You have introduced Persistence Multitenancy in our Pollution Monitoring application on SAP Cloud Platform and initialized the application with the pollution data it needs.

PART 2: IDENTITY MANAGEMENT – ASSIGN APPLICATION ROLES TO USERS IN SAP CP

For Identity management, set up of Identity authentication service tenant is already done for us. To know more how to setup this please refer [here](#). In this part we will be assigning application roles to end users in our consumer account of SAP CP.

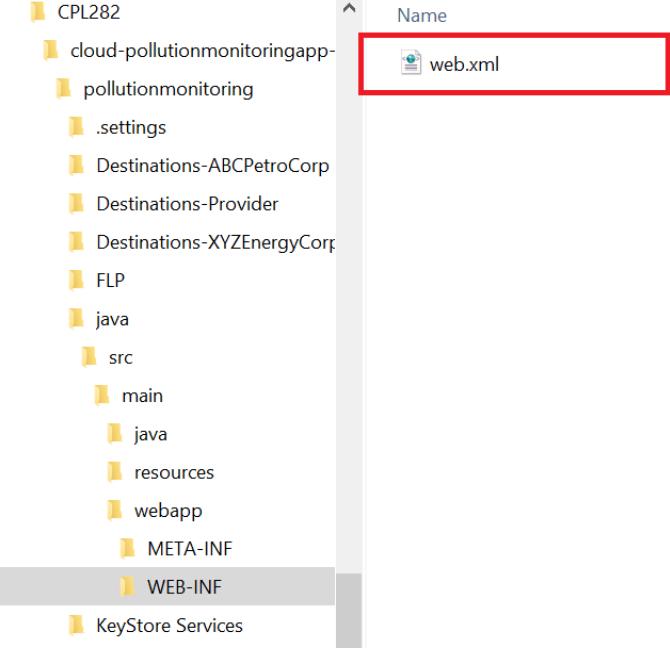
*The Pollution Monitoring multitenant application built by Robert from ITeLO Consulting (Provider), provides two predefined roles **PlantSupervisor** and **AreaManager** which control the authorization of the application and decide what the end-user actually sees in the application.*

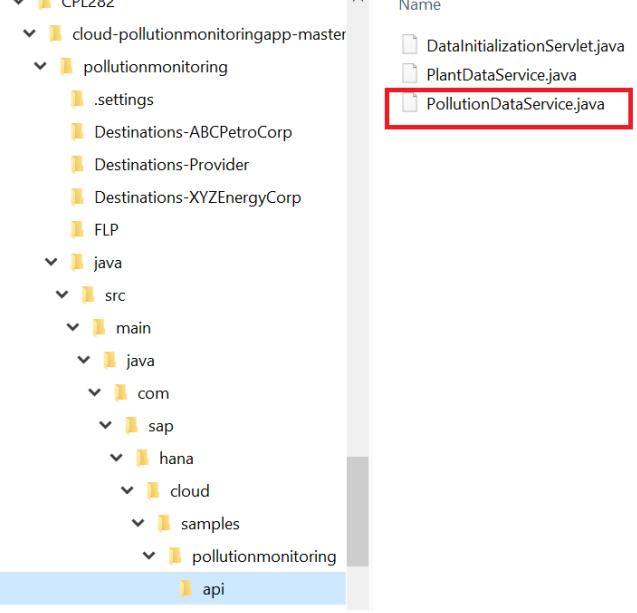
PlantSupervisor: Users who are assigned this role will be able to view the data of **only the plant maintained in his User details (Company Information) in Identity authentication service tenant**.

AreaManager: Users who are assigned this role will be able to view the data of **all plants in his area**.

Let us check out how this is achieved in the code.

Step1: Application logic for Plant Management based upon different roles

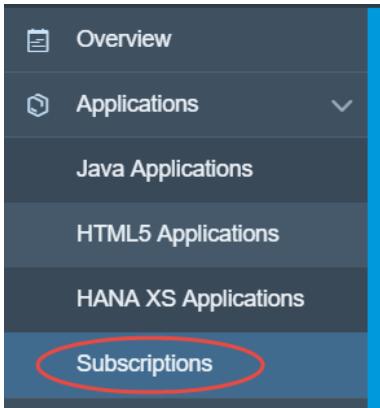
Explanation	Screenshot
<p>1. In the imported project, navigate to /pollutionmonitoring/java/src/main/webapp/WEB-INF/web.xml. Open the web.xml file.</p>	 <p>The screenshot shows a file explorer interface with a tree view of a project structure. The root folder is 'CPL282'. Under it, there's a folder named 'cloud-pollutionmonitoringapp'. Inside 'cloud-pollutionmonitoringapp', there are several subfolders: 'pollutionmonitoring', '.settings', 'Destinations-ABCPetroleumCorp', 'Destinations-Provider', 'Destinations-XYZEnergyCorp', 'FLP', 'java', 'src', 'main', 'java', 'resources', 'webapp', 'META-INF', 'WEB-INF', and 'KeyStore Services'. A red box highlights the 'web.xml' file under the 'WEB-INF' folder. The 'Name' column header is also visible.</p>
<p>2. Check out the two security roles created in the application.</p>	<pre data-bbox="628 1679 1298 1891"> <security-role> <role-name>PlantSupervisor</role-name> </security-role> <security-role> <role-name>AreaManager</role-name> </security-role> </tah->ann> </pre>

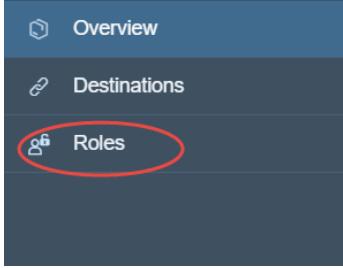
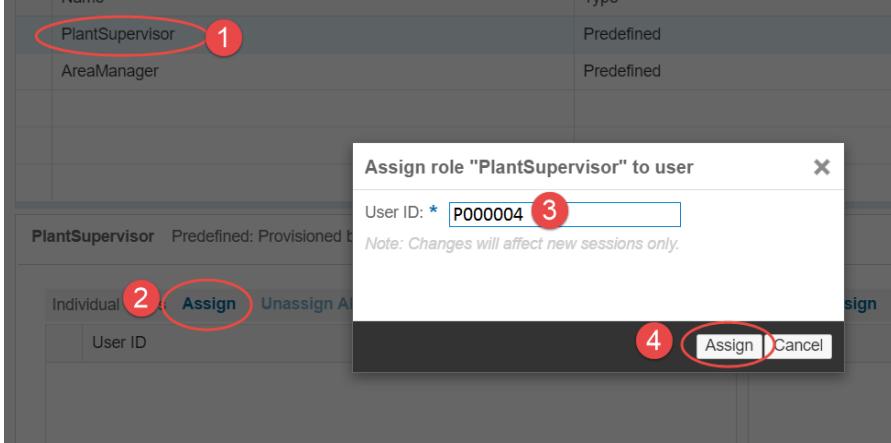
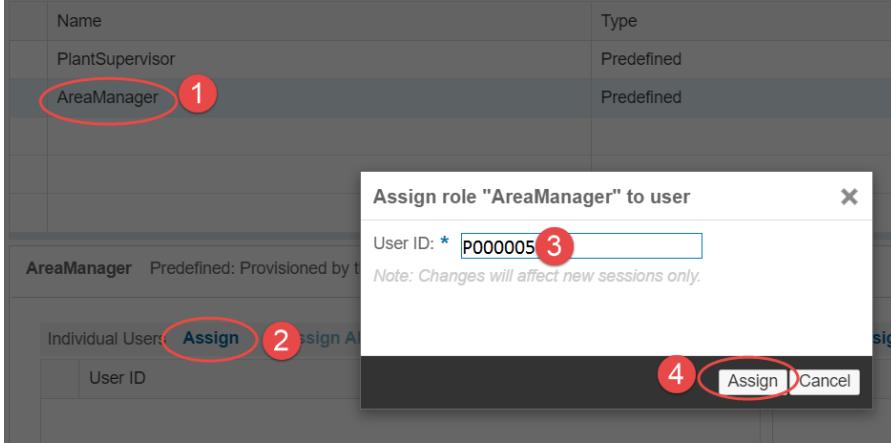
Explanation	Screenshot
<p>3. Navigate to the java file /pollutionmonitoring/java/src/main/java/com/sap/hana/cloud/samples/pollutionmonitoring/api/PollutionDataService.java. Open the java file with notepad++.</p>	 <pre data-bbox="628 369 1265 982"> CPL282 ├── cloud-pollutionmonitoringapp-master │ ├── pollutionmonitoring │ │ ├── .settings │ │ ├── Destinations-ABCPetroleumCorp │ │ ├── Destinations-Provider │ │ ├── Destinations-XYZEnergyCorp │ │ └── FLP │ └── java │ ├── src │ │ ├── main │ │ │ ├── java │ │ │ │ └── com │ │ │ │ └── sap │ │ │ │ └── hana │ │ │ │ └── cloud │ │ │ └── samples │ │ └── pollutionmonitoring └── api └── PollutionDataService.java </pre>
<p>4. Check the method getCompanyPollutionData(). Here we check if the user is an admin (manager). If yes, we show the complete Resultlist, if not we filter by plant_id and show only those results.</p>	<pre data-bbox="628 1003 1489 1257"> boolean isAdmin = isUserAdmin(request); if (isAdmin) { mfPlantList = em.createNamedQuery("Plants").getResultList(); CompanyPollutionData companyPollutionData = prepareCompanyPollutionData(); return companyPollutionData; } else { String plant_id = getPlantId(request); mfPlantList = getPlantPollutionDataById(plant_id); CompanyPollutionData companyPollutionData = prepareCompanyPollutionData(); return companyPollutionData; } </pre>
<p>5. For more clarity, you can also check the method isUserAdmin() where we check the role and decipher if he is an Admin (manager) or not.</p>	<pre data-bbox="628 1299 1489 1616"> private boolean isUserAdmin(HttpServletRequest request) throws PersistenceException, UnsupportedUserAttributeException { String plant_id = getPlantId(request); if (request.getUserPrincipal() != null && request.isUserInRole("AreaManager")) { if (plant_id == null) { // this means he is the admin return true; } else if (request.getUserPrincipal() != null && request.isUserInRole("PlantManager")) { if (plant_id != null) { return false; } } } return false; } </pre>
<p>6. A similar logic is also followed for retrieving the plant data from the on-premise systems .</p>	<pre data-bbox="628 1658 1489 1891"> boolean isAdmin = isUserAdmin(request); if (isAdmin) { return plantInfoService.getPlantsOnPremiseData(); } else { String plant_id = getPlantId(request); return plantInfoService.getPlantOnPremiseData(); } </pre>



Step2: Roles assignment in SAP Cloud Platform cockpit

On the consumer side, Emily will now have to assign these roles to the Area Managers and Plant Supervisors in her company ABC PetroCorp i.e. ABCPlantSupervisor and ABCAreaManager.

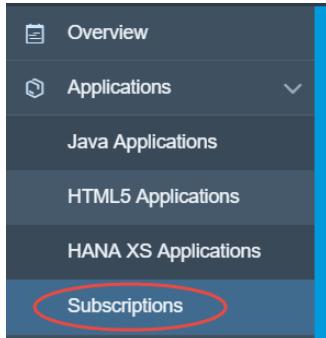
Explanation	Screenshot						
1. In the SAP CP cockpit of the consumer, ABC PetroCorp, navigate to Applications -> Subscriptions .							
2. Click on the pollutionmonitoring subscribed Java application, provided by ITeLO Consulting.	<p>Subscribed Java Applications (All: 1)</p> <table border="1"><thead><tr><th data-bbox="576 1185 666 1206">State</th><th data-bbox="682 1185 829 1206">Provider Account</th><th data-bbox="976 1185 1074 1206">Application</th></tr></thead><tbody><tr><td data-bbox="576 1227 666 1248"></td><td data-bbox="682 1227 927 1248">ITeLO Consulting (Provider)</td><td data-bbox="976 1227 1156 1248">pollutionmonitoring</td></tr></tbody></table>	State	Provider Account	Application		ITeLO Consulting (Provider)	pollutionmonitoring
State	Provider Account	Application					
	ITeLO Consulting (Provider)	pollutionmonitoring					

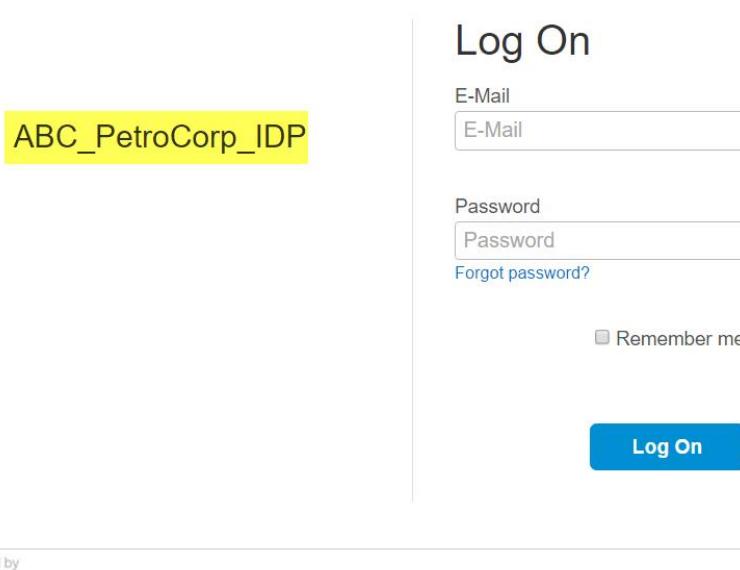
Explanation	Screenshot
<p>3. In the application overview page, navigate to Roles. The list of application roles will be displayed.</p>	
<p>4. Select the PlantSupervisor role and click on Assign. In the pop-up enter the User ID P00004 of the ABCPlantSupervisor and click on Assign.</p>	
<p>5. Similarly, select the AreaManager role and click on Assign. In the pop-up enter the User ID P000005 of the ABCAreaManager (noted in the previous blog after import) and click on Assign.</p>	

Completed! We have successfully mapped the application roles to the corresponding users of **ABC PetroCorp** from Identity authentication service tenant.

Step3: Validation of Identity Management setup

Emily can now check if the setup of the SAP CP account and the application with the corporate IDP, in our case Identity authentication service tenant, works as expected.

Explanation	Screenshot						
<p>1. In the SAP CP cockpit of the consumer, ABC PetroCorp, navigate to Applications -> Subscriptions.</p>	 <p>The screenshot shows the SAP CP cockpit sidebar. The 'Applications' menu is open, displaying four options: Java Applications, HTML5 Applications, HANA XS Applications, and Subscriptions. The 'Subscriptions' option is circled in red.</p>						
<p>2. Click on the link of the subscribed HTML5 application pollutionmonitoringui</p>	<p>Subscribed HTML5 Applications (All: 1)</p> <table border="1"> <thead> <tr> <th data-bbox="589 1143 671 1164">State</th> <th data-bbox="720 1143 883 1164">Provider Account</th> <th data-bbox="1046 1143 1144 1164">Application</th> </tr> </thead> <tbody> <tr> <td data-bbox="589 1185 671 1206"><input checked="" type="checkbox"/></td> <td data-bbox="720 1185 883 1206">[REDACTED]</td> <td data-bbox="1046 1185 1144 1206">pollutionmonitoringui</td> </tr> </tbody> </table> <p>A red oval highlights the 'pollutionmonitoringui' application name in the table.</p>	State	Provider Account	Application	<input checked="" type="checkbox"/>	[REDACTED]	pollutionmonitoringui
State	Provider Account	Application					
<input checked="" type="checkbox"/>	[REDACTED]	pollutionmonitoringui					

Explanation	Screenshot
3. Copy the Application URL link in the Active Version section.	<p>Active Version</p> <p>Active Application Version: 1 Application URL: https://pollutionmonitoringui.dispatcher.ondemand.com</p>
4. Open a browser in <i>Incognito mode/Private mode</i> , paste the URL and click on Go. The login page from the application configuration in SAP Cloud Identity is shown with the correct name.	 <p>The screenshot shows a login form with the following fields:</p> <ul style="list-style-type: none"> E-Mail: ABC_PetroCorp_IDP E-Mail: (empty) Password: (empty) Forgot password? <input type="checkbox"/> Remember me Log On button <p>Powered by SAP Cloud Identity</p>

You should be able to use the User ID of the **AreaManager** or **PlantSupervisor**

Area manager:

User ID Johan
Password Test@12345

PlantSupervisor:

User ID Smith
Password Test@12345

and login to the application and use it without any issues. (*Although, you may find that you do not find any data as the setup is still not complete. This is okay!*)

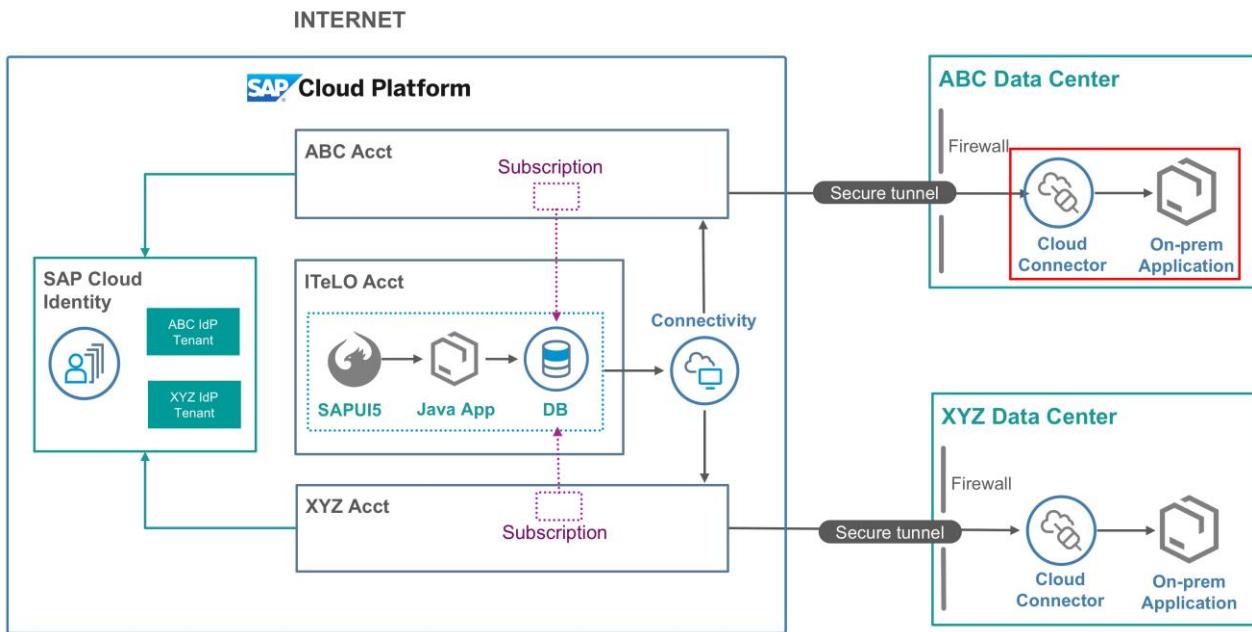
PART 3: ON-PREMISE CONNECTIVITY SERVICE – ON PREMISE SETUP AND CONFIGURE CONNECTIVITY SERVICE

'ITeLO Consulting' is a trusted SAP partner company that specializes in providing solutions for the **Oil and Gas industry**. They have adopted the **SAP Cloud Platform** as the innovation platform to build and run their applications on the cloud

As it is already outlined, ABC PetroCorp –premise service running within their landscape that provide information about its plants – details like no. of workers at each plant, area of the plant etc. The pollution monitoring shows this information in the dashboard.

To achieve this, the on-premise service need to be exposed from their landscape and provided to the cloud application. This can be achieved using the SAP CP Connectivity service

Let's understand how Emily will set up the connectivity between the on-premise services running within ABC PetroCorp network and the application running in the ABC PetroCorp (Consumer)account. Refer to the section highlighted in red below.



Emily will perform the following steps.

Step1: Expose on-premise services securely using SAP Cloud Connector

Step2: Configure connectivity between exposed on-premise services and the cloud application using SAP CP destinations



Step 1: Expose on-premise services securely using SAP Cloud Connector

Note: To make it easy to execute the tutorial, we will simulate the on-premise setup by running a service locally on a Tomcat server installed on our machine. And then use the SAP Cloud Connector installed on our machine to securely expose this service to the cloud.

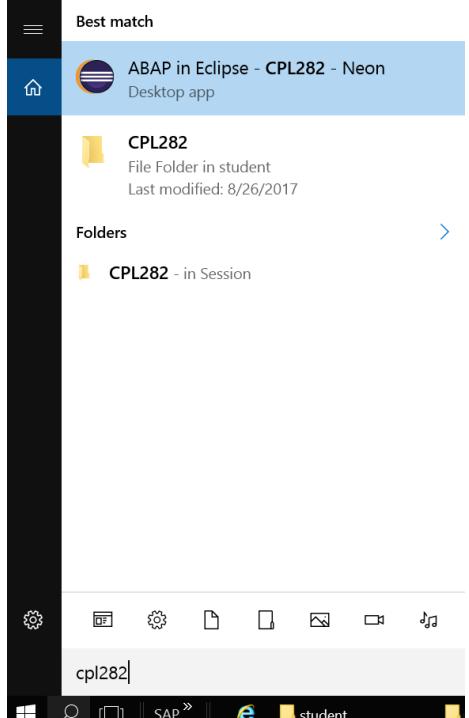
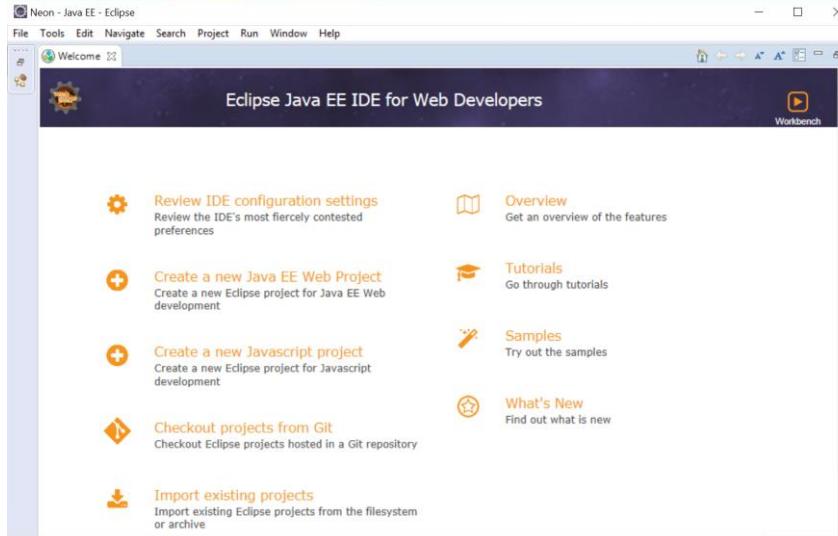
In practice, Emily will work with on-premise services running within the company landscape.

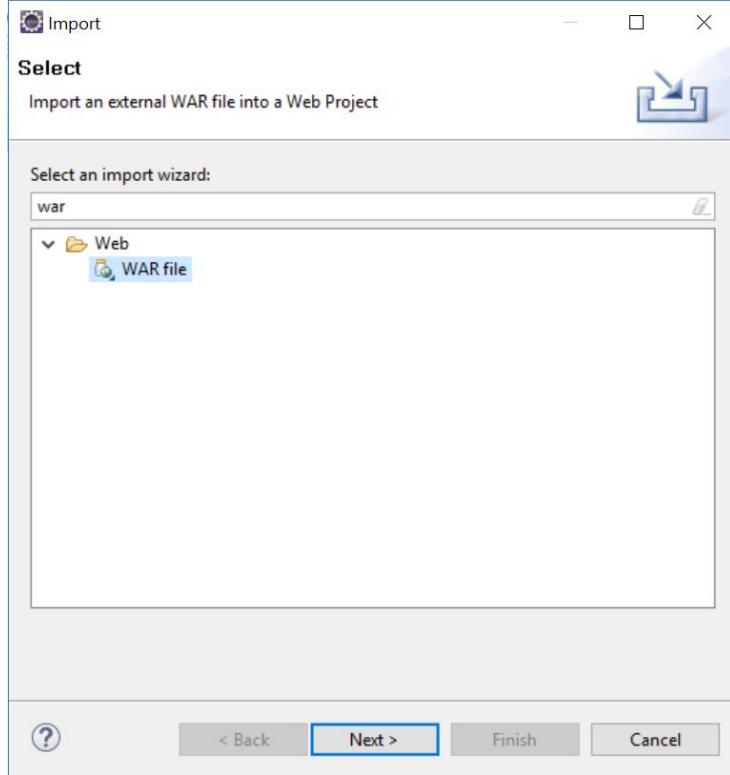
Section 1: Simulate on-premise setup by running a mock service on your machine

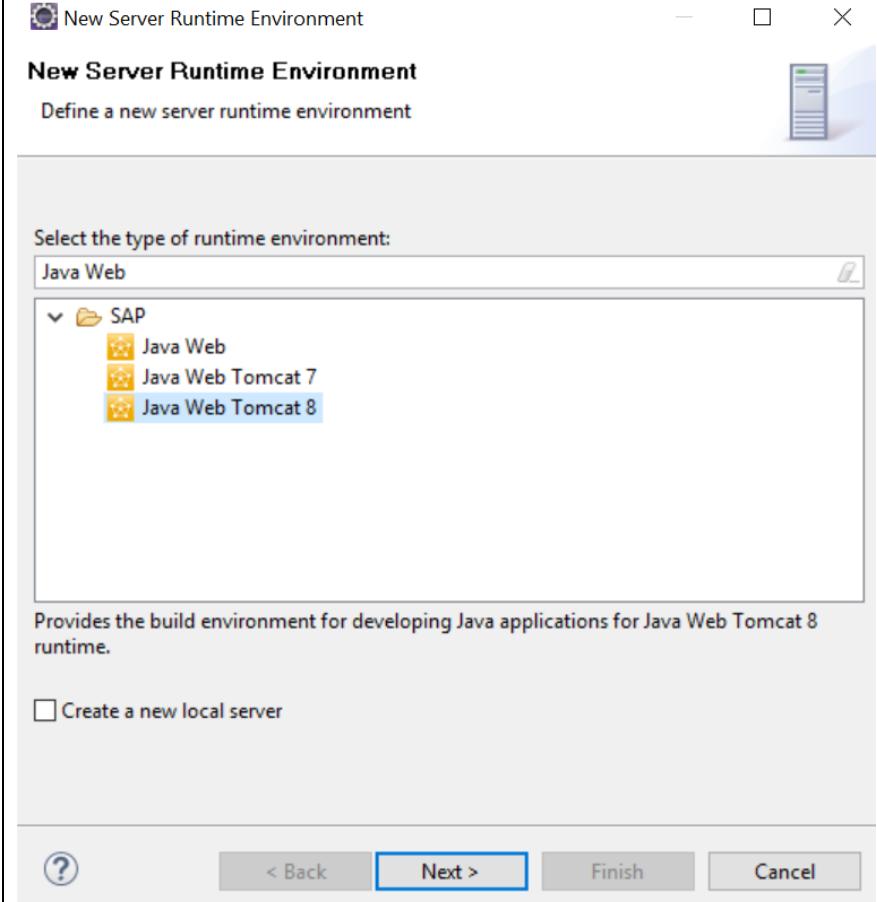
A mock service is available in the Git repository. In this step, you will import, deploy and run this mock service on a local Java Web Tomcat 8 server.

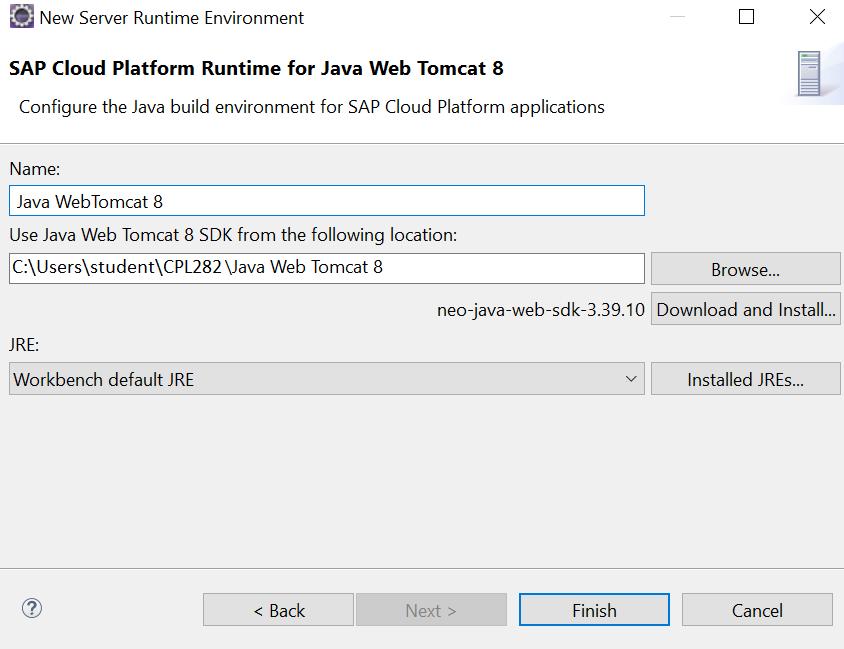
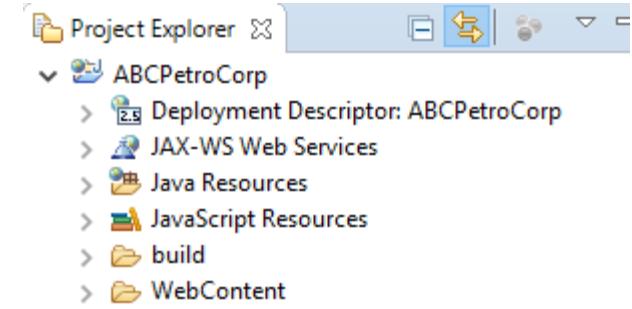
Import project containing mock service into Eclipse

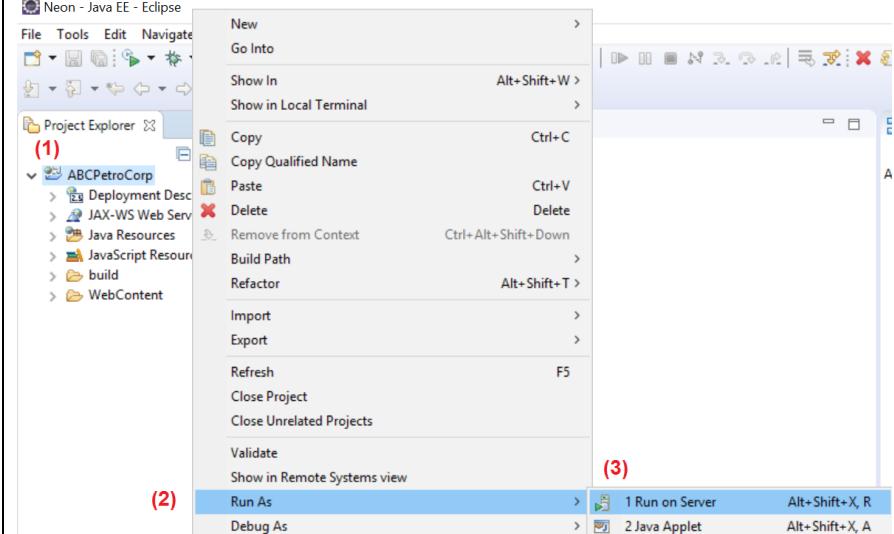
Explanation	Screenshot
<p>1. The mock service has been uploaded as a WAR file under On-Premise Services folder (see screenshot).</p>	<p>The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, displaying a hierarchical tree of project components. A blue selection bar highlights the 'OnPremise Services' folder under the 'cloud-pollutionmonitoringapp-master' project. To the right of the tree, there is a table with two columns: 'Name' and 'Type'. The 'Name' column lists two entries: 'ABCPetroCorp.war' and 'XYZEnergyCorp.war'. The 'Type' column is not visible in the screenshot.</p>

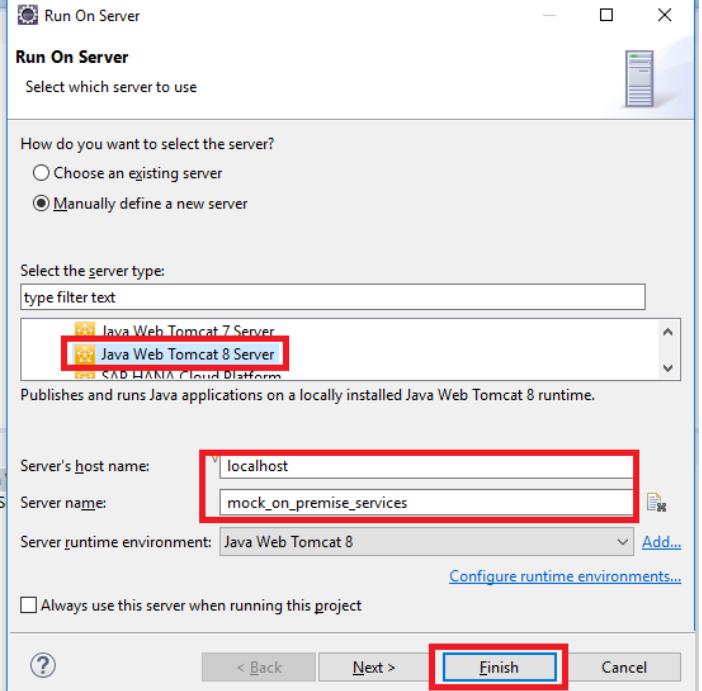
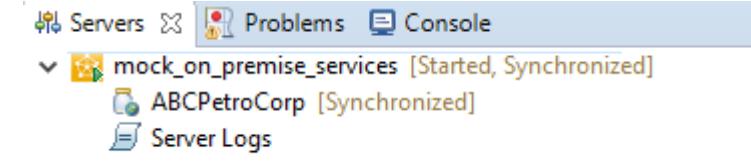
Explanation	Screenshot
<p>2. Open the eclipse by searching cpl 282 in the Search windows and select the Best match ABAP in eclipse-CPL282-Neon as shown in the screenshot.</p>	
<p>3. This will open the eclipse in Java EE perspective with a Welcome View. Close the Welcome view.</p>	

Explanation	Screenshot
<p>4. Click on File>Import option in and search war in an search bar to select an import wizard. Click on Next.</p>	

Explanation	Screenshot
<p>5. In the WAR Import dialog, provide the following info:</p> <p>WAR file: File location of the ABC PetroCorp.war file</p> <p>Web project: ABCPetroCorp</p> <p>6. To select Target runtime click on New and search Java Web in the search bar.</p> <p>7. Select Java Web Tomcat 8 server under SAP Folder and click on Next.</p>	

Explanation	Screenshot
<p>8. Browse for Java Web Tomcat 8 SDK stored on your system.</p> <p>Note: The Zip file of JAVA Web Tomcat 8 will be available at Studentshare/CPL282/JavaWebTomcat8.zip</p> <p>Copy this file and paste into C:\Users\student\CPL282 and extract the content.</p> <p>In case if it is not available please download from this site. https://tools.hana.ondemand.com/#cloud under SAP Cloud Platform Neo Environment section.</p> <p>Click on Finish.</p>	
<p>9. After a successful import, in the Project Explorer view, the project structure should look like this.</p>	
<p>Deploy and run the mock service on local Java Web Tomcat 8 server</p>	

Explanation	Screenshot
<p>10. Deploy imported on-premise service on to local server (Java Web Tomcat 8).</p> <p>In the Project Explorer, right click on ABCpetroCorp node. Select Run As>Run on Server.</p>	 <p>The screenshot shows the Eclipse Neon interface with the Java EE - Eclipse perspective. In the Project Explorer view, there is a single project node named "ABCPetroCorp" with a red circle labeled "(1)" next to it. A context menu is open over this node, with the "Run As" option highlighted by a red circle labeled "(2)". Under the "Run As" option, the "Run on Server" option is also highlighted with a red circle labeled "(3)". Other options visible in the menu include New, Go Into, Show In, Copy, Paste, Delete, Remove from Context, Build Path, Refactor, Import, Export, Refresh, Close Project, Close Unrelated Projects, Validate, Show in Remote Systems view, Run As, and Debug As.</p>

Explanation	Screenshot
<p>11. In the Run on Server dialog, choose “Manually define a new server”.</p> <p>Enter the following details: Server's host name: localhost Server name: mock_on_premise_services Click on Finish.</p>	
<p>12. Once the deployment has been done, you should see the following structure under Servers view.</p>	
<p>13. This mock service can be accessed at the URL http://localhost:8080/ABCPetrolCorp/ Note: It could be possible that your server does not run on port 8080. In that case please check the port number of your server and execute http://localhost:<your port number>/ABCPetrolCorp.</p>	<p>This would return:</p> <pre>{"plants": [{"numberOfWorkers": "7543", "country": "CANADA", "crudeOilCapacity": "7857 gallons", "plantId": "101", "location": "New BrunsWick", "plantArea": "1212 Sq feet", "plantName": "LPG Production"}, {"numberOfWorkers": "10000", "country": "CANADA", "crudeOilCapacity": "20000 gallons", "plantId": "102", "location": "Alberta", "plantArea": "5252 Sq feet", "plantName": "Oilfield Production"}, {"numberOfWorkers": "8000", "country": "CANADA", "crudeOilCapacity": "18000 gallons", "plantId": "103", "location": "British Columbia", "plantArea": "4846 Sq feet", "plantName": "Gas Processing and Compression Plant"}], "companyName": ["ABC PetroCorp, CANADA"]}</pre>

Section 2: Expose the mock service using SAP Cloud Connector

The ABC Petro Corp mock service can be exposed to the application running on SAP CP in a secure and easy way using [SAP Cloud Connector](#).

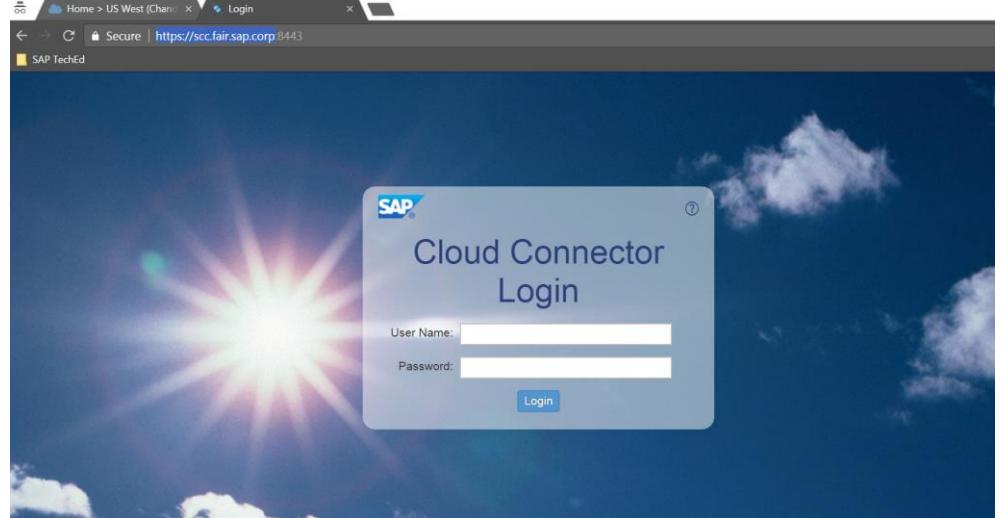
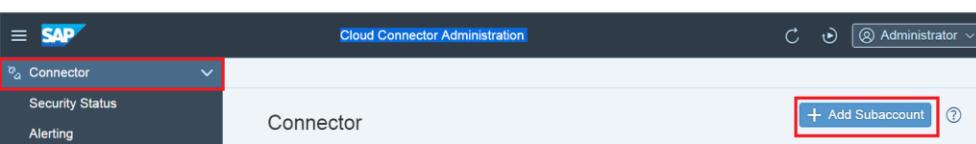
For executing this tutorial easily, the SAP Cloud Connector will be installed on your local machine. In the actual scenario, Emily will install and configure the Cloud Connector within the company landscape.

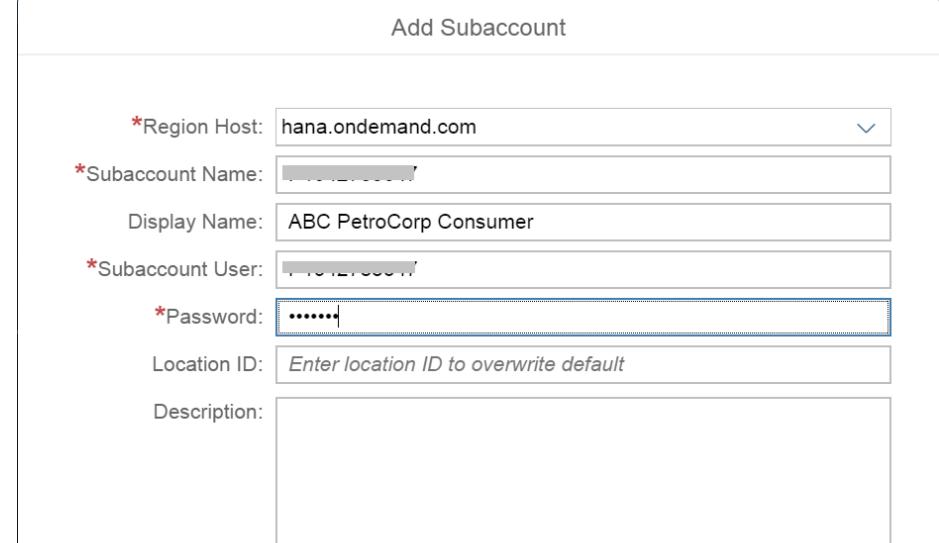
Pre-requisites:

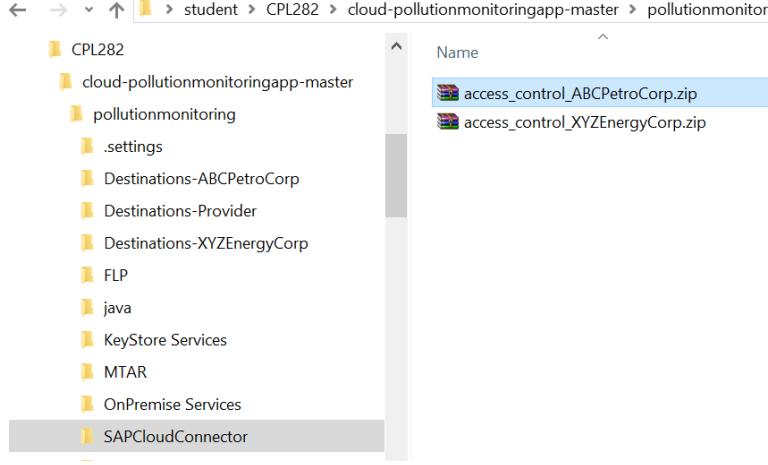
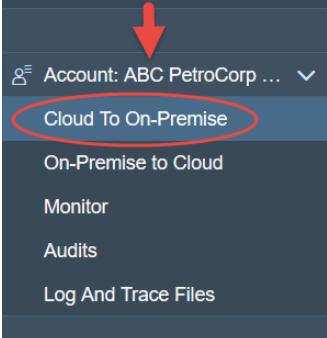
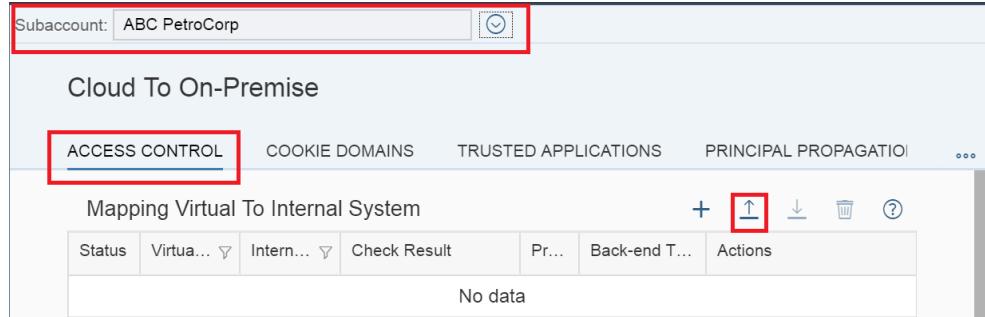
1. You have SAP HANA Cloud Connector [installed](#)
2. You have performed the [initial configuration](#) on SAP Cloud Connector

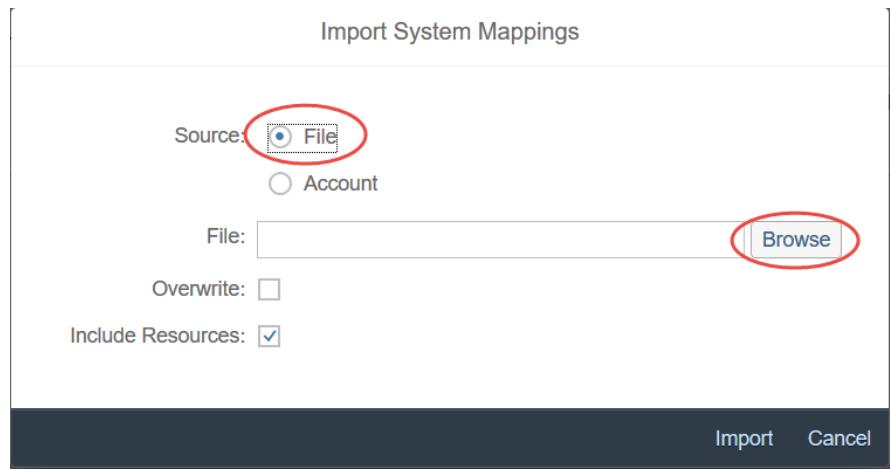
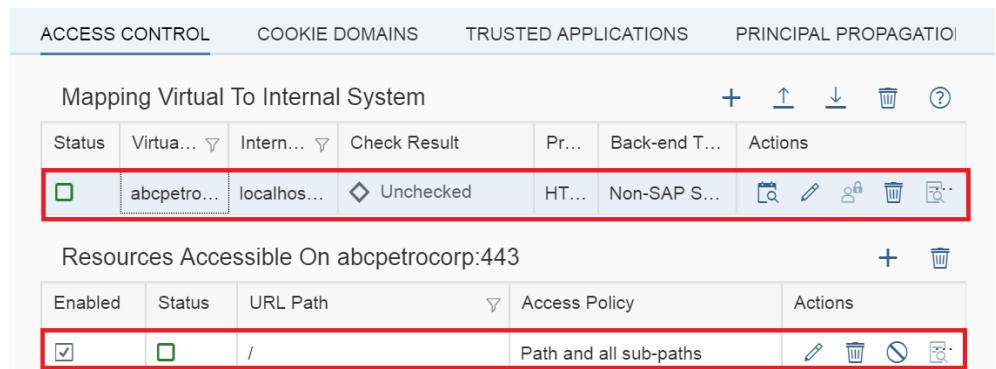
Configuring Access Control for consumer account

The internal on-premise system will be exposed using a virtual system. This virtual system then will be used in the HTTP destination configuration in SAP SAP Cloud Platform in the next step (Step 2). More info on configure access control can be found [here](#).

Explanation	Screenshot
<p>1. Log in to the SAP Cloud connector with URL <a href="https://<hostname>:<port>">https://<hostname>:<port> https://scc.fair.sap.corp:8443</p> <p>User Name: Administrator Password: welcome</p>	
<p>2. Add the SAP CP account to which you want to expose the on-premise mock services. Here we are connecting to the ABC PetroCorp (Consumer) Navigate to the Connector tab and click on Add Subaccount</p>	

Explanation	Screenshot												
<p>3. Enter details of the ABC PetroCorp (Consumer) account and click Save.</p> <p>Region Host: hana.ondemand.com</p> <p>Subaccount Name: <Subaccount Name> To find subaccount name, switch to your SAP Cloud Platform Cockpit and navigate to overview page of your ABC PetroCorp subaccount. You will find subaccount name ABC PetroCorp(P1942783xxx) under Subaccount Information</p> <p>Subaccount User: P1942783xxx Password: Welcome17</p> <p>Click on Save.</p> <p>Use HTTPS Proxy Host: proxy Port: 8080</p>													
	<p style="text-align: center;">HTTPS Proxy</p> <p style="text-align: center;">Host: proxy Port: 8080</p>												
<p>4. Once added, it will be listed in the Subaccount Dashboard.</p>	<p style="text-align: center;">Subaccount Dashboard</p> <table border="1" data-bbox="592 1460 1564 1559"> <thead> <tr> <th>Status</th> <th>Subaccou... ▾</th> <th>Display N... ▾</th> <th>Location ID ▾</th> <th>Region Host ▾</th> <th>Actions</th> </tr> </thead> <tbody> <tr> <td>◆</td> <td>redacted</td> <td>redacted</td> <td></td> <td>hana.ondemand....</td> <td> </td> </tr> </tbody> </table>	Status	Subaccou... ▾	Display N... ▾	Location ID ▾	Region Host ▾	Actions	◆	redacted	redacted		hana.ondemand....	
Status	Subaccou... ▾	Display N... ▾	Location ID ▾	Region Host ▾	Actions								
◆	redacted	redacted		hana.ondemand....									

Explanation	Screenshot
<p>5. Next step is to configure access control. There has to be a mapping created between the local system and the virtual system.</p> <p>This has been made available as access_control_<<consumer name>>.zip file under the SAPCloudConnector folder.</p> <p>You can import this file to configure access control in the SAP Cloud connector as described in the upcoming steps.</p>	
<p>6. Navigate to Cloud To On-Premise section and under the ABC PetroCorp Account.</p>	
<p>7. Check that the Subaccount dropdown shows the correct account and go to ACCESS CONTROL tab.</p> <p>Click on Import corresponding under the Mapping Virtual to Internal System section as shown in the screenshot.</p>	

Explanation	Screenshot
<p>8. This opens Import System Mappings dialog. Select Source as File and choose Browse. Select the location of the <code>access_control_ABCPetroCorp.zip</code> file and click on Import.</p>	
<p>9. This will import the access control setting on to your SAP Cloud connector.</p> <p>Note: The <code>access_control_ABCPetroCorp.zip</code> file assumes port number 8080 for the on-premise service. In case your mock service runs on different port, then Edit the mapping with the correct port number.</p>	
<p>10. Check that the access control has been configured correctly as shown in the screenshot.</p>	
<p>11. In the SAP CP cloud cockpit for the ABC PetroCorp (P19442783xxx) account, navigate to the</p>	

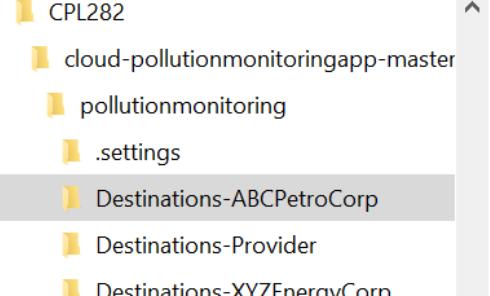
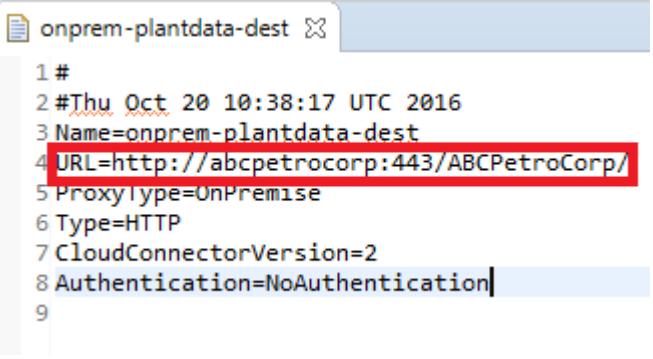
Explanation	Screenshot								
<p>Connectivity>Cloud Connectors tab.</p> <p>You would see the status of the SAP Cloud Connector as shown in the screenshot.</p>	<p>SAP Cloud Platform Cockpit</p> <p>Subaccount: ABC PetroCorp(P1942783712) - Cloud Connectors</p> <p>Connected</p> <p>Master Instance</p> <p>Connector ID: 812DEAB13FB511E7C155F7BA0A40D916 Connected since: 24.10.2017 06:10:35 *Initiated by: P1942783712 Version: 2.10.1 Java Version: 1.8.0_131 (SAP AG) High Availability: inactive</p> <p>Force Disconnect</p> <p>Exposed Back-End Systems</p> <table border="1"> <thead> <tr> <th>Host</th> <th>Protocol</th> <th>Back-End Type</th> <th>Resources</th> </tr> </thead> <tbody> <tr> <td>abcpetrocorp:443</td> <td>HTTP</td> <td>Non-SAP System</td> <td>Available</td> </tr> </tbody> </table>	Host	Protocol	Back-End Type	Resources	abcpetrocorp:443	HTTP	Non-SAP System	Available
Host	Protocol	Back-End Type	Resources						
abcpetrocorp:443	HTTP	Non-SAP System	Available						

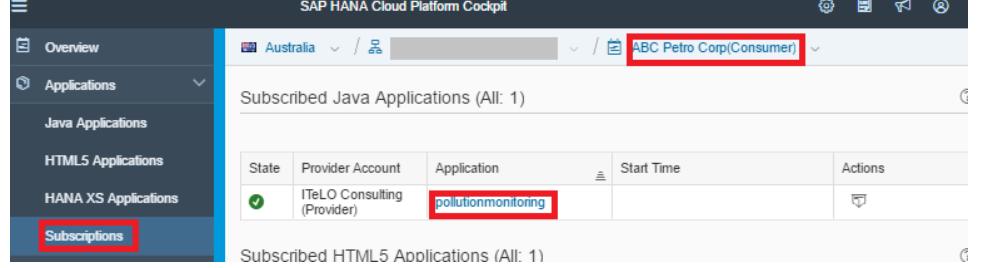
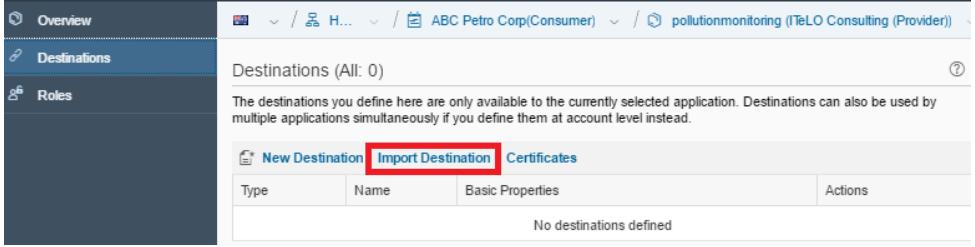
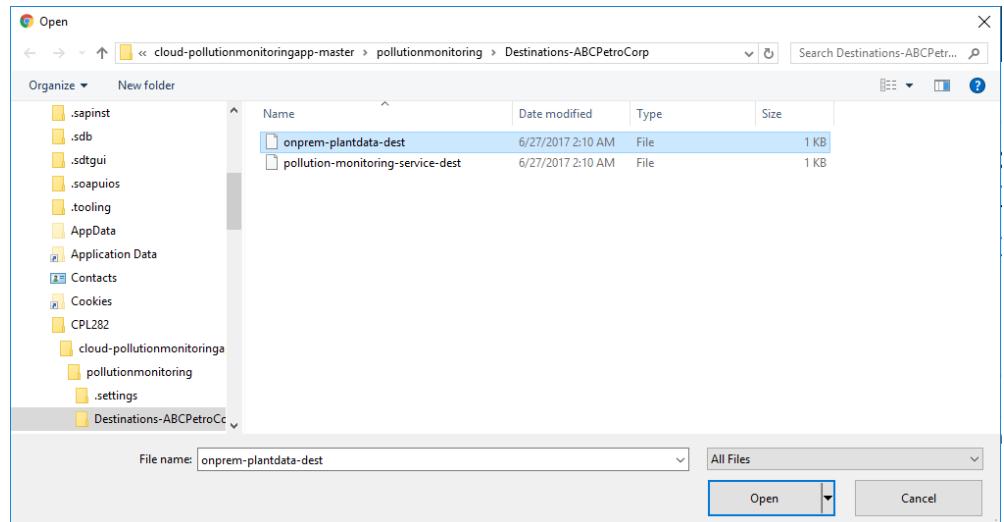
With this step, the mock **on-premise service** have been exposed to the **ABC PetroCorp** (Consumer) account.

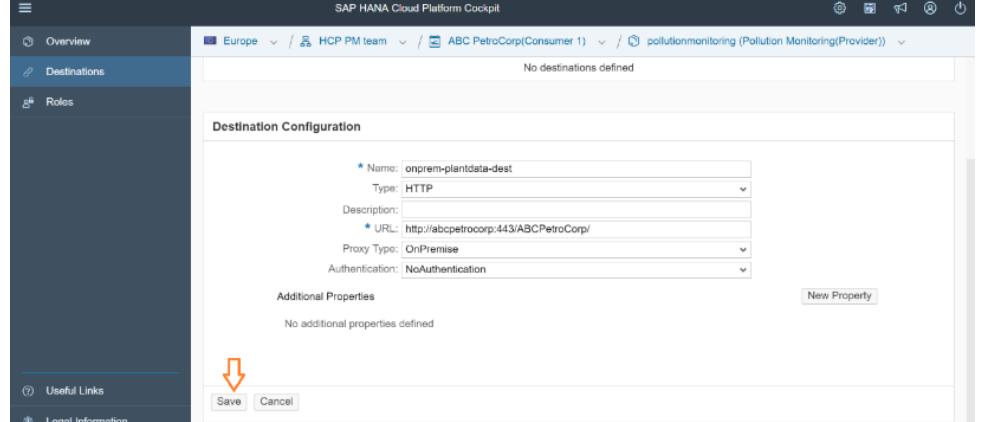
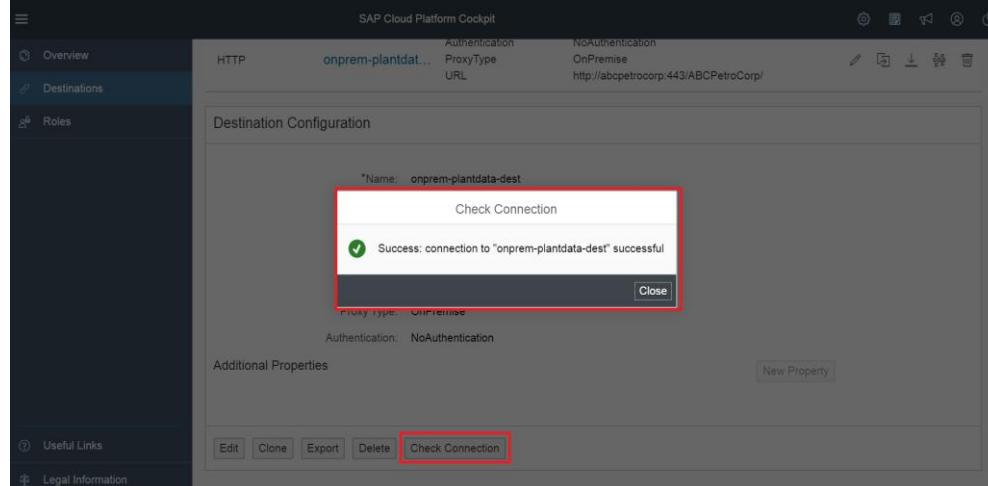
Step 2: Configure connectivity between exposed on-premise services and the cloud application using SAP CP destinations

In this step, the Java cloud application on SAP CP will connect to the exposed mock on-premise services through HTTP destinations in SAP CP.

Section 1: Configure Destinations from SAP CP

Explanation	Screenshot
<p>1. There is a destination file onprem-plandata-dest file that is available under the folder Destinations-ABCPetrolCorp</p>	 <p>The screenshot shows a tree view of SAP CP destinations. The 'Destinations-ABCPetrolCorp' folder is highlighted with a gray background. Other visible folders include 'CPL282', 'cloud-pollutionmonitoringapp-master', 'pollutionmonitoring', '.settings', 'Destinations-Provider', and 'Destinations-XV7FenergCorporation'.</p>
<p>2. This destination points to the mock on-premise service URL which has been already exposed to the ABC PetroCorp Subaccount in the previous step.</p> <p>This destination needs to be uploaded to the ABC PetroCorp (Consumer) Subaccount as shown in the steps below.</p>	 <pre> 1 # 2 #Thu Oct 20 10:38:17 UTC 2016 3 Name=onprem-plandata-dest 4 URL=http://abcpetrocorp:443/ABCPetrolCorp/ 5 ProxyType=OnPremise 6 Type=HTTP 7 CloudConnectorVersion=2 8 Authentication=NoAuthentication 9 </pre> <p>The 'URL' line in the configuration file is highlighted with a red rectangle.</p>

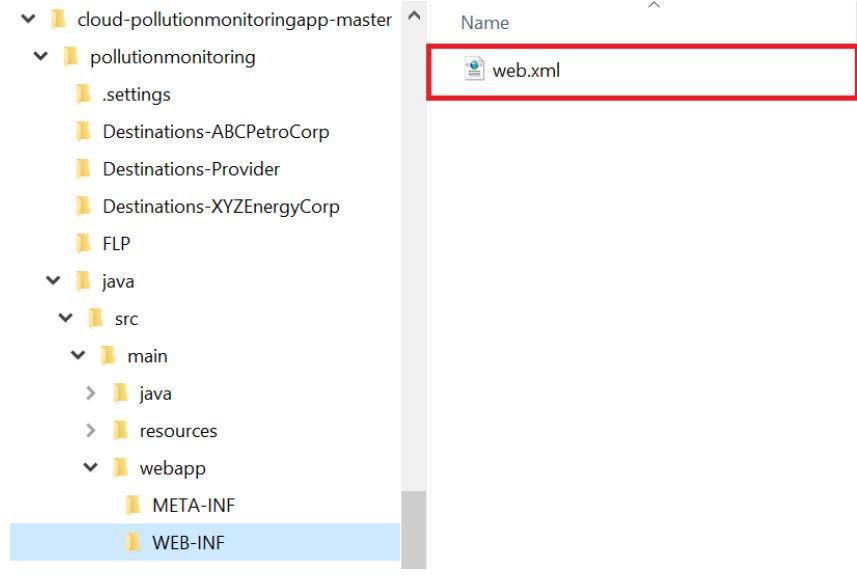
Explanation	Screenshot
<p>3. Navigate to Subscriptions pane of ABC PetroCorp (Consumer) Subaccount and click on pollutionmonitoring Java application subscription</p>	 <p>The screenshot shows the SAP HANA Cloud Platform Cockpit interface. The left sidebar has a 'Subscriptions' tab highlighted with a red box. The main content area shows a table for 'Subscribed Java Applications'. There is one entry: 'ITeLO Consulting (Provider)' with the application name 'pollutionmonitoring' also highlighted with a red box.</p>
<p>4. Navigate to Destinations pane of your subscribed pollutionmonitoring Java application and click on Import Destination.</p>	 <p>The screenshot shows the Destinations pane for the 'pollutionmonitoring' application. The 'Import Destination' button is highlighted with a red box. The pane displays a table with columns 'Type', 'Name', and 'Basic Properties'.</p>
<p>5. Select the file location of onprem-plandata-dest file from your Project.</p>	 <p>The screenshot shows a file explorer dialog with the path 'cloud-pollutionmonitoringapp-master > pollutionmonitoring > Destinations-ABCpetroCorp'. The file 'onprem-plandata-dest' is selected and highlighted with a blue selection bar. The 'File name:' field contains 'onprem-plandata-dest'.</p>

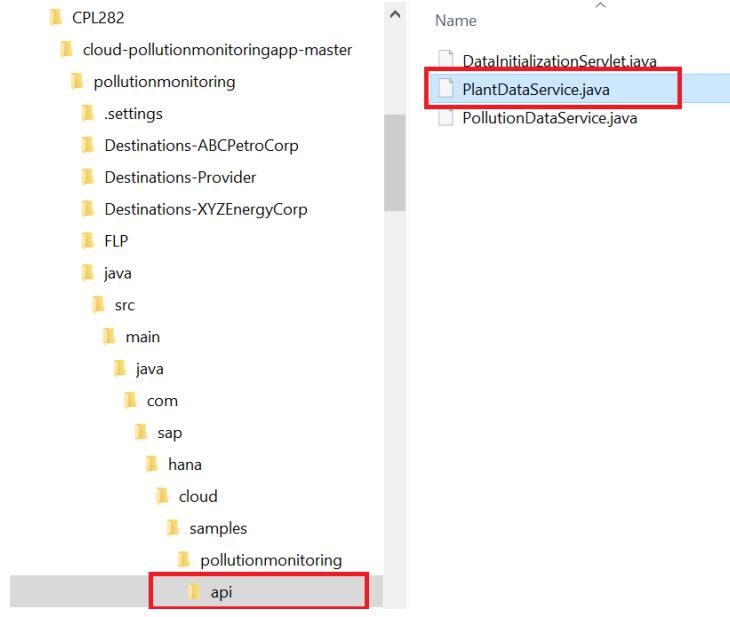
Explanation	Screenshot
<p>6. The destination would be added in your Subaccount as shown in the screenshot click on Save.</p>	 <p>The screenshot shows the SAP HANA Cloud Platform Cockpit interface. The left sidebar has 'Destinations' selected. The main area is titled 'Destination Configuration' for a destination named 'onprem-plantdata-dest'. The configuration includes fields for Name (onprem-plantdata-dest), Type (HTTP), URL (http://abcpetrocpr:443/ABCPetrcorp/), Proxy Type (OnPremise), and Authentication (NoAuthentication). Below the form, it says 'Additional Properties' with 'No additional properties defined'. At the bottom are 'Save' and 'Cancel' buttons, with a red arrow pointing to the 'Save' button.</p>
<p>7. Verify that on click on Check Connection you will get following message in a dialog</p> <p>Success: connection to 'onprem-Plant-dest' successful.</p>	 <p>The screenshot shows the SAP Cloud Platform Cockpit interface. The left sidebar has 'Destinations' selected. The main area shows a destination configuration for 'onprem-plantdata-dest' with fields for HTTP, Authentication (NoAuthentication), and URL (http://abcpetrocpr:443/ABCPetrcorp/). Below the form, there are buttons for Edit, Clone, Export, Delete, and a red-highlighted 'Check Connection' button. A modal dialog box is open, titled 'Check Connection', displaying the message 'Success: connection to "onprem-plantdata-dest" successful.' with a checkmark icon. The entire dialog box is outlined with a red border.</p>

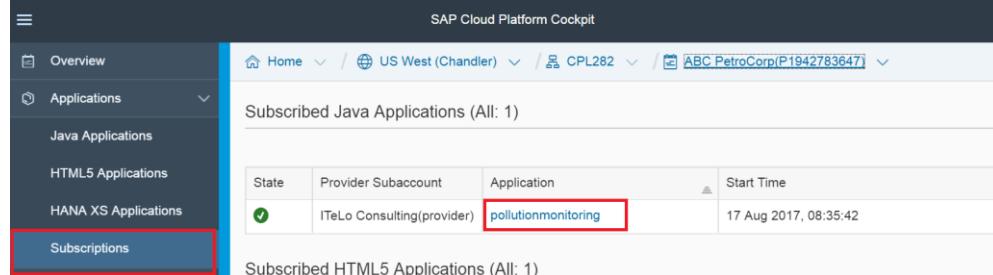
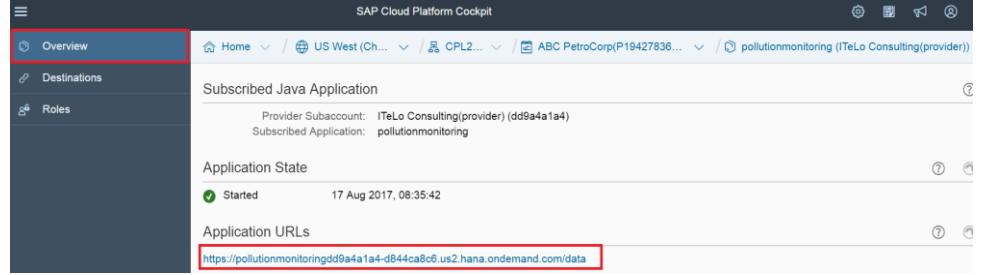


Section 2: Access the on-premise service from the cloud Java application

If the on-premise connectivity setup is working fine, then the **on-premise** service would be accessible via the pollutionmonitoring Java application.

Explanation	Screenshot
<p>1. Navigate to /pollutionmonitoring/java/src/main/webapp/WEB-INF/web.xml. Open the web.xml file with notepad++</p>	
<p>2. In the Java application, the destination onprem-plantdata-dest has been configured in the web.xml file.</p>	<p>web.xml</p> <pre data-bbox="576 1122 1319 1333"><resource-ref> <res-ref-name>onprem-plantdata-dest</res-ref- name> <res- type>com.sap.core.connectivity.api.http.HttpDesti nation</res-type> </resource-ref></pre>

Explanation	Screenshot
<p>3. Navigate to /pollutionmonitoring/java/src/main/java/com/sap/hana/cloud/samples/pollutionmonitoring/api to edit PlantDataService.java with notepad++.</p>	 <pre>CPL282 ├── cloud-pollutionmonitoringapp-master │ ├── pollutionmonitoring │ │ ├── .settings │ │ ├── Destinations-ABCPetroleumCorp │ │ ├── Destinations-Provider │ │ ├── Destinations-XYZEnergyCorp │ │ ├── FLP │ │ ├── java │ │ │ ├── src │ │ │ └── main │ │ │ ├── java │ │ │ │ └── com │ │ │ │ └── sap │ │ │ └── hana │ │ └── cloud │ └── samples └── pollutionmonitoring └── api</pre>
<p>4. This destination onprem-plantdata-dest is being looked up in the PlantDataService.java. Once the destination look up has been done, all calls from the Java code to the mock service are triggered via this destination.</p>	<p>PlantDataService.java</p> <pre>ConnectivityConfiguration configuration = (ConnectivityConfiguration) ctx .lookup("java:comp/env/connectivityConfiguration"); // Get destination configuration for "destinationName" DestinationConfiguration destConfiguration = configuration.getConfiguration("onprem-plantdata- dest");</pre>

Explanation	Screenshot								
<p>5. We will test this service call from the Java application. in the ABC PetroCorp subaccount, navigate to the Subscriptions tab click on the Subscribed java application pollutionmonitoring.</p>	 <table border="1" data-bbox="813 530 1569 593"> <thead> <tr> <th data-bbox="813 530 862 551">State</th><th data-bbox="862 530 992 551">Provider Subaccount</th><th data-bbox="992 530 1123 551">Application</th><th data-bbox="1123 530 1569 551">Start Time</th></tr> </thead> <tbody> <tr> <td data-bbox="813 551 862 593">✓</td><td data-bbox="862 551 992 593">ITeLo Consulting(provider)</td><td data-bbox="992 551 1205 593">pollutionmonitoring</td><td data-bbox="1205 551 1569 593">17 Aug 2017, 08:35:42</td></tr> </tbody> </table>	State	Provider Subaccount	Application	Start Time	✓	ITeLo Consulting(provider)	pollutionmonitoring	17 Aug 2017, 08:35:42
State	Provider Subaccount	Application	Start Time						
✓	ITeLo Consulting(provider)	pollutionmonitoring	17 Aug 2017, 08:35:42						
<p>6. From the Overview pane, copy the pollutionmonitoring Java Application URL.</p>	 <p>Provider Subaccount: ITeLo Consulting(provider) (dd9a4a1a4) Subscribed Application: pollutionmonitoring</p> <p>Application State Started 17 Aug 2017, 08:35:42</p> <p>Application URLs https://pollutionmonitoringdd9a4a1a4-d844ca8c6.us2.hana.ondemand.com/data</p>								
<p>7. Append '/api/v1/plantdta' to the the URL and run the service call in a browser. The URL will look like:</p> <pre data-bbox="258 1263 551 1396">https://pollutionmonitoring<provider account id>-<consumer account id>.<<landscape_host>>/data/api/v1/plantdata</pre> <p>The plant details returned from the mock on-premise services (invoked by the Java application) are shown in the browser.</p>	 <pre data-bbox="576 1291 1569 1354"> [{"plants": [{"id": "101", "name": "New Brunswick", "area": "1212 Sq feet", "capacity": "7857 gallons", "workers": "7543", "country": "CANADA", "location": "New Brunswick"}, {"id": "102", "name": "Alberta", "area": "5252 Sq feet", "capacity": "20000 gallons", "workers": "10000", "country": "CANADA", "location": "Alberta"}, {"id": "103", "name": "British Columbia", "area": "4844 Sq feet", "capacity": "18000 gallons", "workers": "8000", "country": "CANADA", "location": "British Columbia"}], "company": "ABC PetroCorp, CANADA"} </pre>								

That's it! The pollution monitoring application in SAP CP is connected to the on-premise services running within the **ABC PetroCorp** network. The important point to note here is that the connectivity was established at the subscription(tenant) level.

In the next part, we will look at connecting the SAP CP's HTML5 application to the Java service running on the SAP Cloud Platform

PART 4: CONFIGURE A CONNECTIVITY SERVICE AND TEST THE MULTITENANT APPLICATION

Step 1: Configure connectivity on the consumer account (to service running on SAP CP)

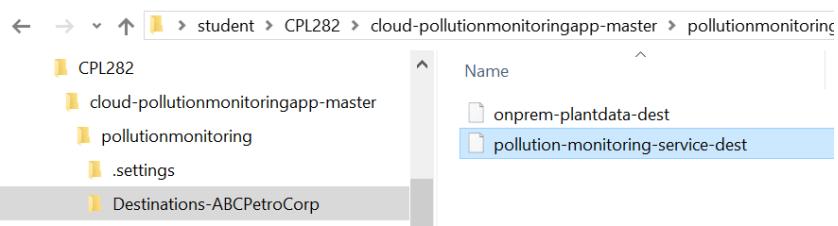
Step 2: Test the Pollution Monitoring dashboard by login to HTML5 Application

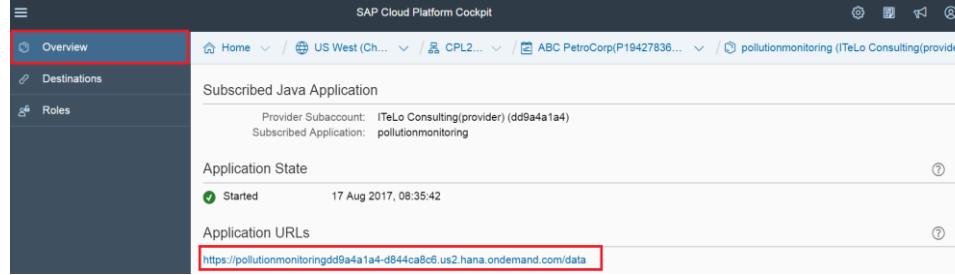
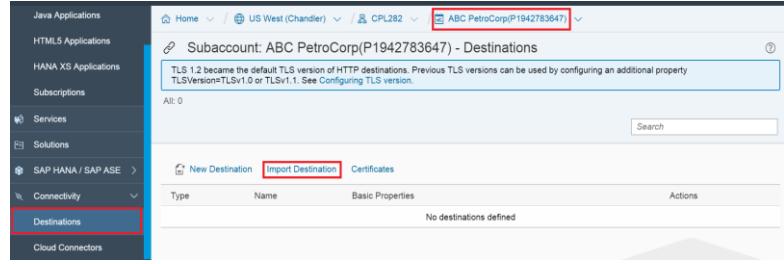
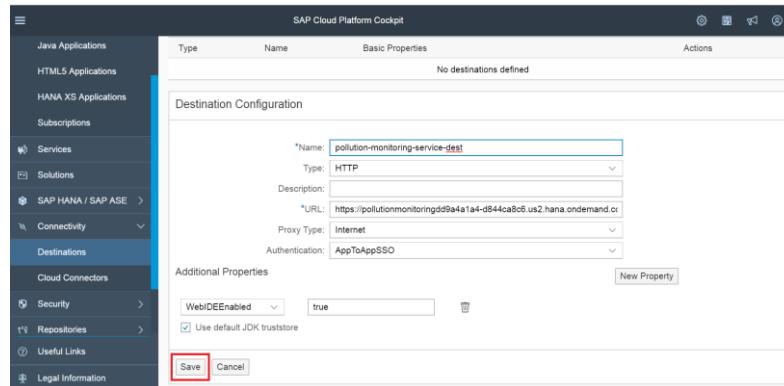
Let's look at these steps in detail.

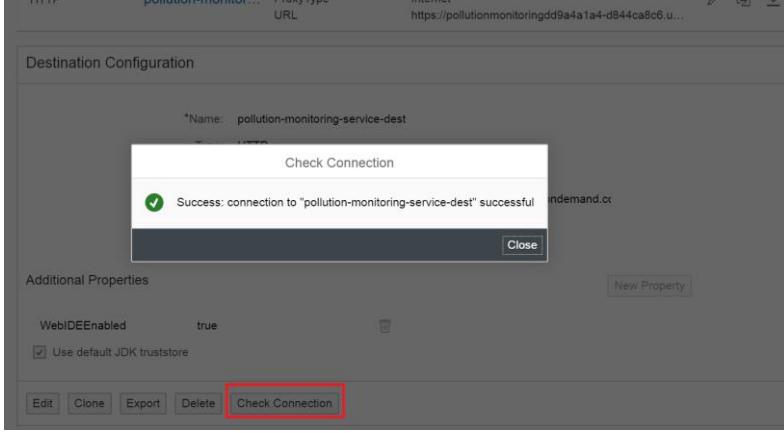
Step 1: Configure connectivity on the consumer account (to service running on SAP CP)

In consumer account, HTML5 application uses services for displaying plant and pollution data from Java application. Hence, there is need to configure destination in the consumer account that points to subscribed Java application. This destination will be consumed by the HTML5 application running in same consumer account.

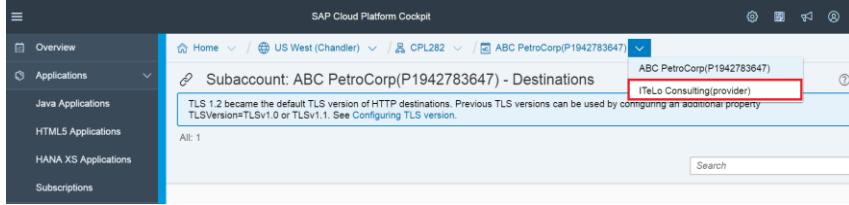
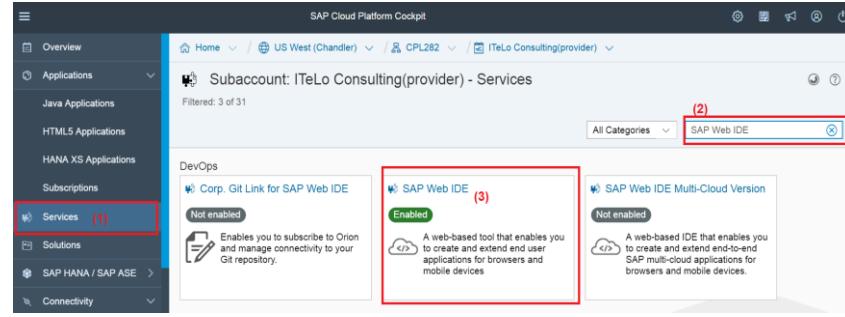
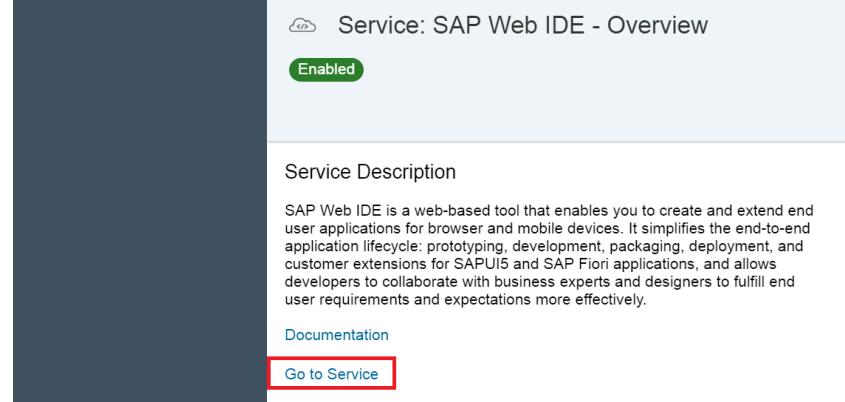
Let's understand how Emily will configure this destination that points to the subscribed pollution monitoring Java application running in **ABC PetroCorp** Subaccount.

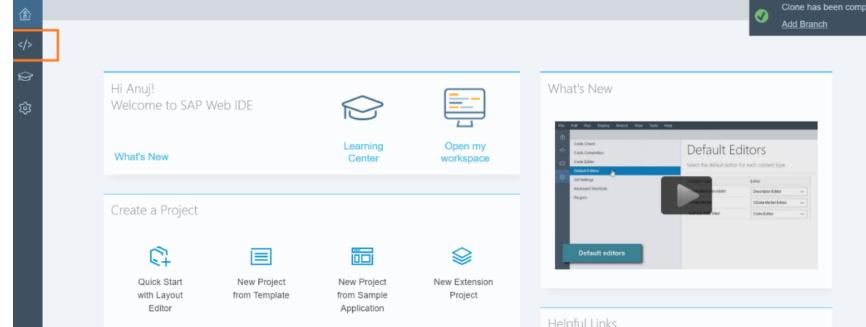
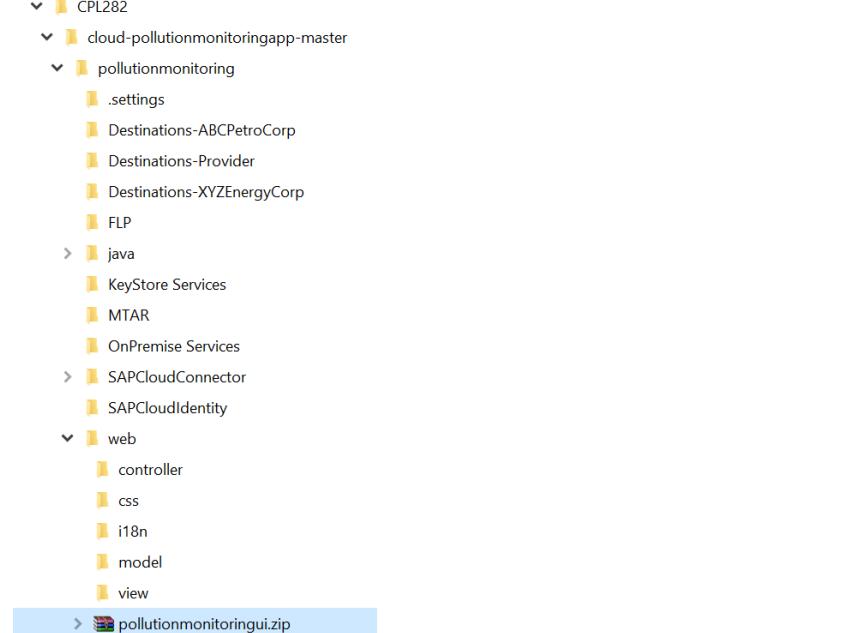
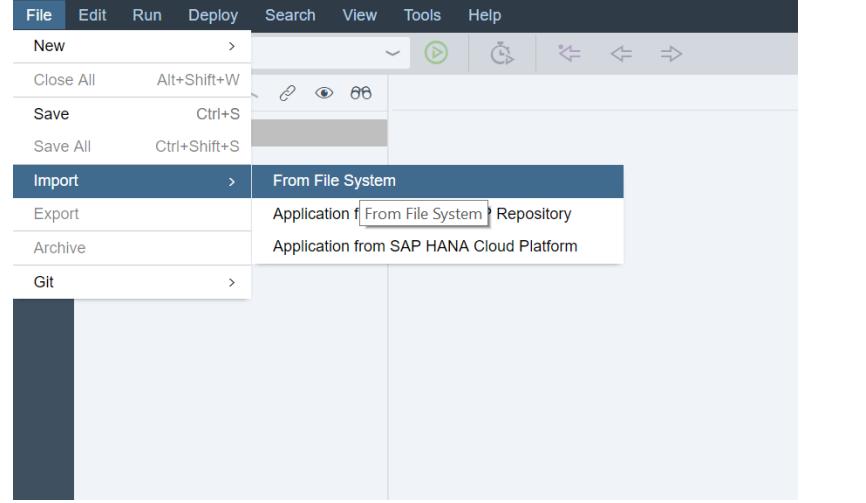
Explanation	Screenshot
<p>1. There is a destination file pollution-monitoring-service-dest file that is available under the folder Destinations-ABC_PetroCorp.</p>	
<p>2. Here, the destination URL points to the pollutionmonitoring Java application which ABC PetroCorp Subaccount is subscribed to.</p> <p>Since this destination is to be consumed by HTML5 Application, WebIDEEnabled is true and Authentication is AppToAppSSO (Application to Application Single Sign on) URL needs to be copied from your Java application subscribed URL as mention in the next step.</p>	<pre data-bbox="698 1517 1474 1855"> pollution-monitoring-service-dest 1# 2#Thu Oct 20 10:38:23 UTC 2016 3Name=pollution-monitoring-service-dest 4# URL = <<Insert your java app subscription URL>> 5ProxyType=Internet 6Type=HTTP 7CloudConnectorVersion=2 8WebIDEEnabled=true 9Authentication=AppToAppSSO </pre>

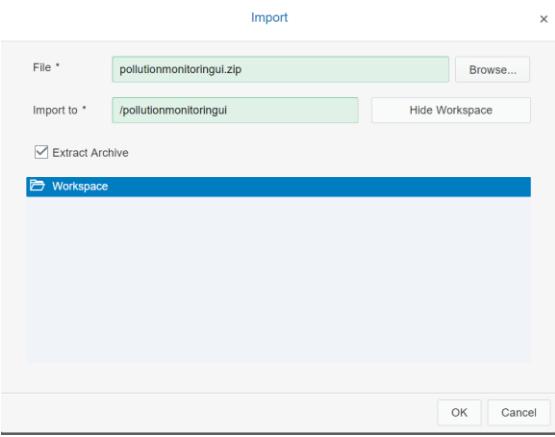
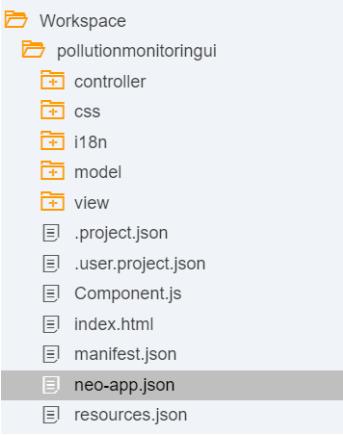
Explanation	Screenshot
<p>3. Copy the pollutionmonitoring Java application subscribed URL from Application URLs section in overview pane.</p>	
<p>4. Paste this URL into the URL parameter and remove # in the pollution-monitoring-service-dest file.</p>	<pre># #Thu Oct 20 10:38:23 UTC 2016 Name=pollution-monitoring-service-dest URL =https://pollutionmonitoringdd9a4a1a4-d844ca8c6.us2.hana.ondemand.com/data ProxyType=Internet Type=HTTP CloudConnectorVersion=2 WebIDEEnabled=true Authentication=AppToAppSSO</pre>
<p>5. Navigate to Destinations pane of your ABC PetroCorp Subaccount click on 'Import Destination'.</p>	
<p>6. Select the file location of pollution-monitoring-service-dest file. The destination would be added in your account as shown in the screenshot click on Save.</p>	

Explanation	Screenshot
<p>7. Verify that on click on Check Connection you will get following message in a dialog</p> <p>Success: connection to 'pollution-monitoring-service-dest' successful.</p>	 <p>The screenshot shows the SAP Web IDE interface with a 'Destination Configuration' dialog open. The dialog title is 'Check Connection' and the message inside says 'Success: connection to "pollution-monitoring-service-dest" successful'. Below the dialog, there is a section titled 'Additional Properties' with a checkbox 'Use default JDK truststore' checked. At the bottom of the dialog, there are several buttons: 'Edit', 'Clone', 'Export', 'Delete', and 'Check Connection', with 'Check Connection' being highlighted by a red box.</p>

Let's understand that how **pollutionmonitoringui** HTML5 application consume subscribed **pollutionmonitoring** Java application. For this you need to import the **pollutionmonitoringui** into the SAP Web IDE workspace.

Explanation	Screenshot
<p>1. Switch to the ITeLo Consulting (provider) Subaccount through dropdown in the breadcrumb.</p>	
<p>2. Navigate to Services pane of ITeLo Consulting (Provider) Subaccount and search for SAP Web IDE service and click on SAP Web IDE.</p>	
<p>3. Click on Go to service.</p>	

Explanation	Screenshot
<p>4. SAP Web IDE Editor opens in new Tab of the same browser.</p> <p>5. Click on the “</>” Development pane.</p>	 <p>The screenshot shows the SAP Web IDE interface. The left sidebar has a 'Development' icon highlighted with a red box. The main area displays a welcome message 'Hi Anuj! Welcome to SAP Web IDE' and a 'Create a Project' section with four options: 'Quick Start with Layout Editor', 'New Project from Template', 'New Project from Sample Application', and 'New Extension Project'. On the right, there's a 'What's New' section and a 'Helpful Links' section.</p>
<p>6. Navigate the location of HTML5 app pollutionmonitoringui.zip file defined under the folder web.</p>	 <p>The screenshot shows the SAP Web IDE file tree. The 'pollutionmonitoringui.zip' file is selected and highlighted with a blue box. The tree structure includes: <ul style="list-style-type: none"> CPL282 cloud-pollutionmonitoringapp-master <ul style="list-style-type: none"> pollutionmonitoring <ul style="list-style-type: none"> .settings Destinations-ABCPetroCorp Destinations-Provider Destinations-XYZEnergyCorp FLP java <ul style="list-style-type: none"> KeyStore Services MTAR OnPremise Services SAPCloudConnector SAPCloudIdentity web <ul style="list-style-type: none"> controller css i18n model view </p>
<p>7. Import the pollutionmonitoringui HTML5 app using File->Import->From File System.</p>	 <p>The screenshot shows the SAP Web IDE 'File' menu open. The 'Import' option is selected and highlighted with a blue box. A dropdown menu is open under 'Import', showing 'From File System' as the selected option. Other options include 'Application' (with 'From File System' and 'Repository' sub-options), 'Archive', and 'Git'.</p>

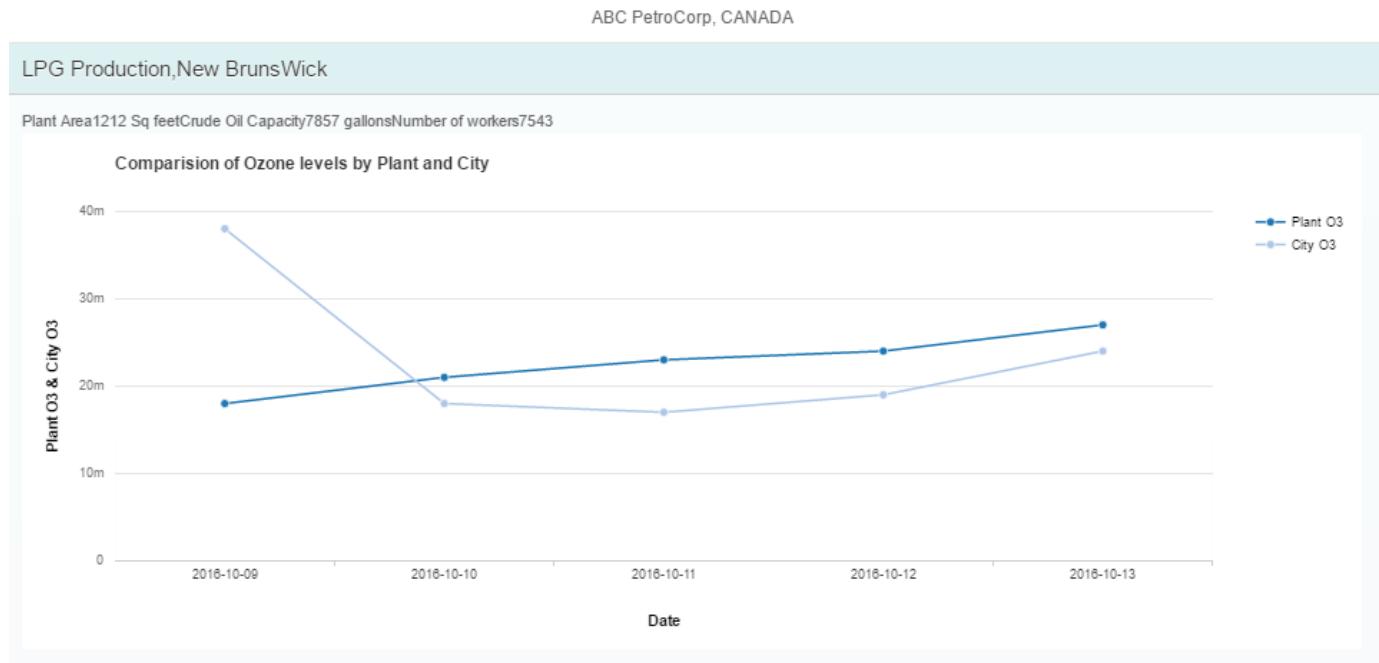
Explanation	Screenshot
<p>8. Click on Browse and choose the pollutionmonitoringui.zip file from the location mention above.</p> <p>Import to /pollutionmonitoringui folder under Workspace. Select Extract Archive check Box. Click on OK.</p>	
<p>9. HTML5 app Pollutionmonitoringui imported into the SAP WEB IDE workspace and looks like this.</p>	
<p>10. Open the neo-app.json file.</p>	
<p>11. The destination pollution-monitoring-service-dest is defined in the neo-app.json file(between line number 13 to 20) which invokes pollutionmonitoring Java application.</p>	<pre data-bbox="617 1432 1373 1734">neo-app.json 13 { 14 "path": "/data", 15 "target": { 16 "type": "destination", 17 "name": "pollution-monitoring-service-dest" 18 }, 19 "description": "Oil and Gas Plant list" 20 },</pre>



Step 2. Test the Pollution Monitoring dashboard by login to HTML5 Application

Emily login to **pollutionmonitoringui HTML5 Application** by following instructions of [Part 2-Step3 \(Validating the Identity Management setup\)](#) and test whether Pollution Monitoring dashboard

- displays plant and pollution data
- compare pollution (ozone) level by plant and city
- authorize users to display UI based upon their roles (all plants or single plant information)



PART 5: FIORI LAUNCHPAD – CONFIGURING TILE FOR MULTI-TENANT APPLICATION

Robert has created the SAP Fiori Launchpad (FLP) site for the pollution monitoring dashboard application running on the provider account. He has also exported this site for his customers. More information on this can be found [here](#).

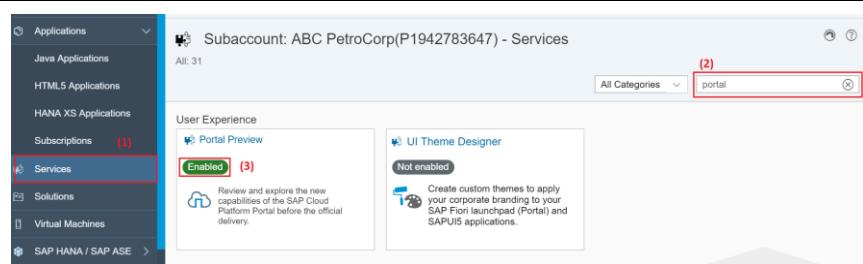
Emily now can import and run this SAP FLP site into ABC PetroCorp subaccount and will perform the following steps:

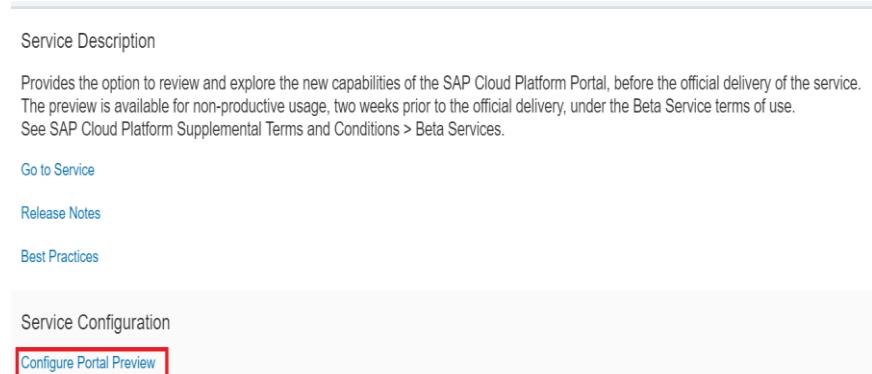
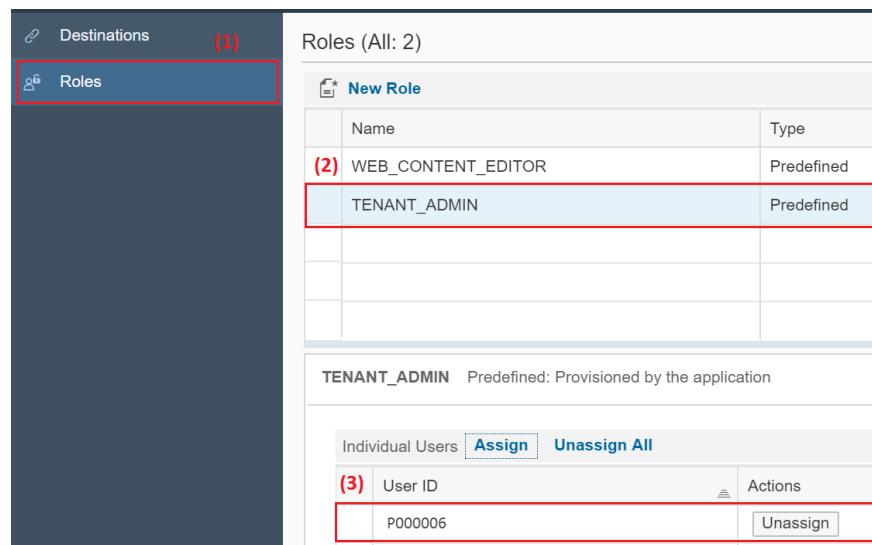
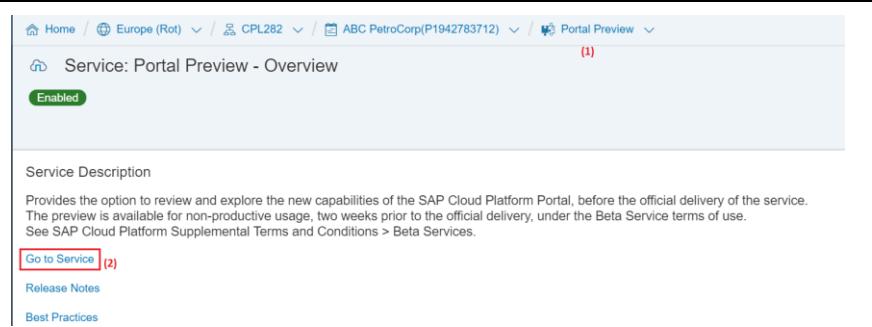
Step 1: Login to SAP CP Portal Service and Import SAP Fiori Launchpad content into ABC PetroCorp Subaccount

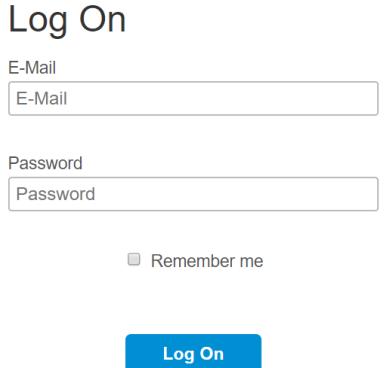
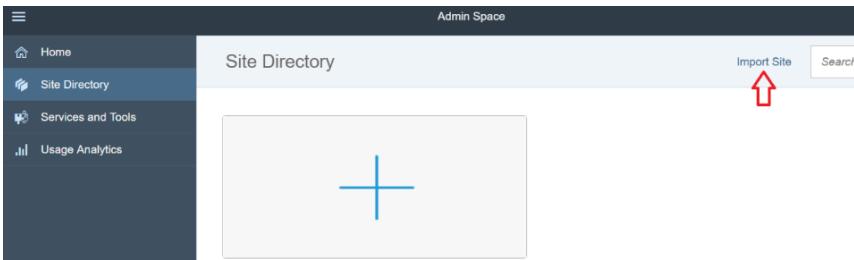
Step 2: Publish and Launch SAP Fiori Launchpad site from the ABC PetroCorp Subaccount

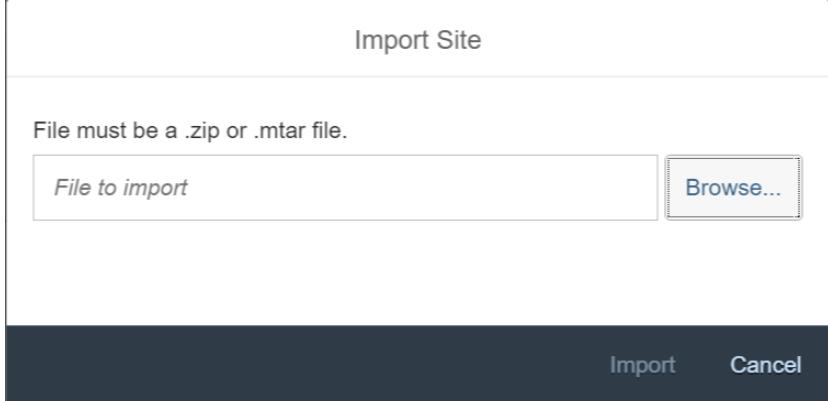
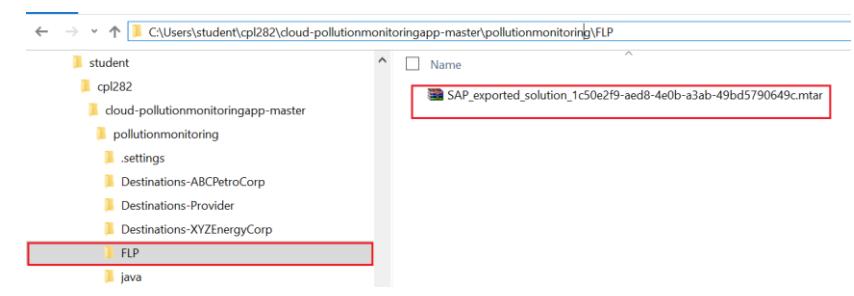
Note: Currently SAP FLP does not support subscriptions. It is not possible to subscribe to an FLP site running in the provider account but for the sake of completeness we are importing SAP FLP site **containing configuration of HTML5 application** into the consumer Subaccount.

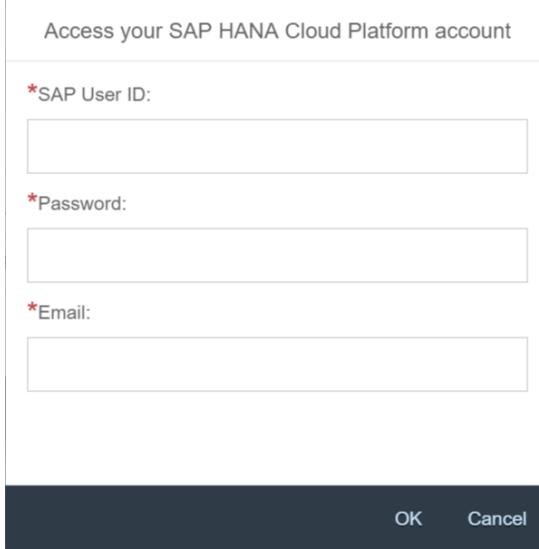
Step1: Login to SAP CP Portal Service and Import SAP Fiori Launchpad content into ABC PetroCorp Subaccount

Explanation	Screenshot
<p>1. In the ABC PetroCorp (P1942783xxx) Subaccount Click on Services to view the complete list of services offered by the SAP Cloud Platform for your account. Search for portal under All Categories to locate the Portal Preview. Ensure that the service is Enabled for your account.</p>	

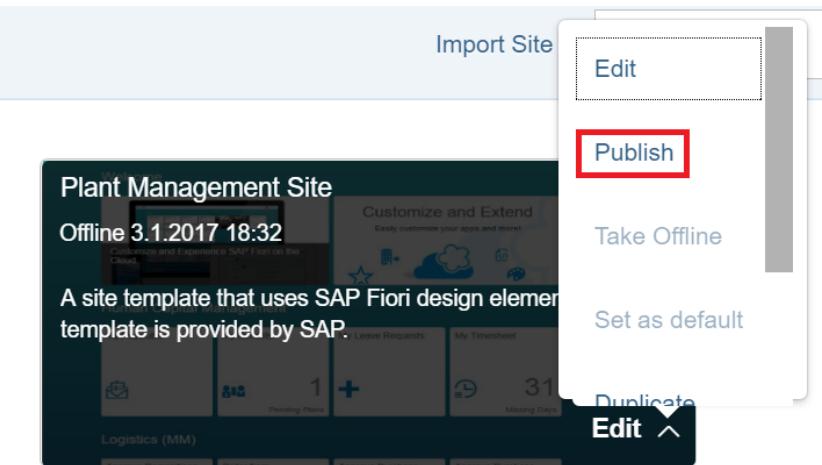
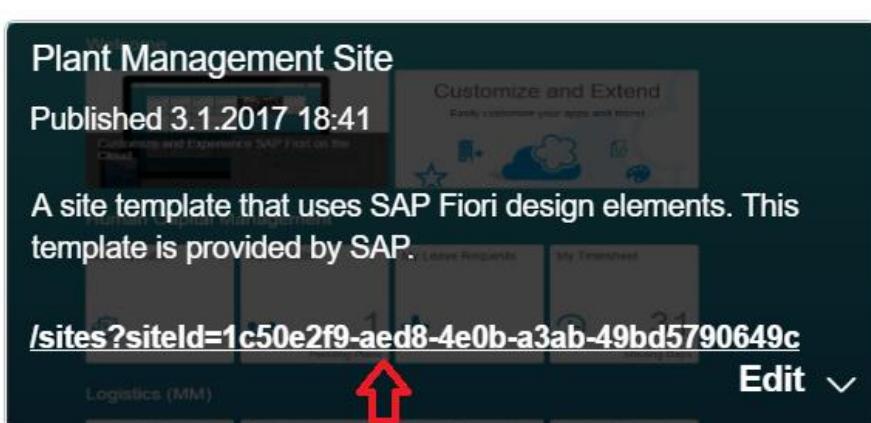
Explanation	Screenshot														
<p>2. This will open the Service Description page for Portal service.</p> <p>Click on the Configure Portal Preview to assign the predefined TENANT_ADMIN role for administrative permissions of the SAP FLP.</p>	 <p>Service Description</p> <p>Provides the option to review and explore the new capabilities of the SAP Cloud Platform Portal, before the official delivery of the service. The preview is available for non-productive usage, two weeks prior to the official delivery, under the Beta Service terms of use. See SAP Cloud Platform Supplemental Terms and Conditions > Beta Services.</p> <p>Go to Service</p> <p>Release Notes</p> <p>Best Practices</p> <p>Service Configuration</p> <p>Configure Portal Preview</p>														
<p>3. Navigate to Roles pane and Select TENANT_ADMIN Role And assign to Emily User_ID in the IDP that is P000006.</p>	 <p>Destinations (1)</p> <p>Roles (2)</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>(2) WEB_CONTENT_EDITOR</td> <td>Predefined</td> </tr> <tr> <td>TENANT_ADMIN</td> <td>Predefined</td> </tr> </tbody> </table> <p>TENANT_ADMIN Predefined: Provisioned by the application</p> <table border="1"> <thead> <tr> <th colspan="2">Individual Users</th> <th>Assign</th> <th>Unassign All</th> </tr> </thead> <tbody> <tr> <td>(3) User ID</td> <td>P000006</td> <td></td> <td>Unassign</td> </tr> </tbody> </table>	Name	Type	(2) WEB_CONTENT_EDITOR	Predefined	TENANT_ADMIN	Predefined	Individual Users		Assign	Unassign All	(3) User ID	P000006		Unassign
Name	Type														
(2) WEB_CONTENT_EDITOR	Predefined														
TENANT_ADMIN	Predefined														
Individual Users		Assign	Unassign All												
(3) User ID	P000006		Unassign												
<p>4. Click on Portal to switch back to the Service Description Page.</p> <p>Click on Go to Service to launch SAP Cloud Platform, Portal Preview in a new browser tab.</p>	 <p>Home / Europe (Rot) / CPL282 / ABC PetroCorp(P1942783712) / Portal Preview (1)</p> <p>Service: Portal Preview - Overview</p> <p>Enabled</p> <p>Service Description</p> <p>Provides the option to review and explore the new capabilities of the SAP Cloud Platform Portal, before the official delivery of the service. The preview is available for non-productive usage, two weeks prior to the official delivery, under the Beta Service terms of use. See SAP Cloud Platform Supplemental Terms and Conditions > Beta Services.</p> <p>Go to Service (2)</p> <p>Release Notes</p> <p>Best Practices</p>														

Explanation	Screenshot
<p>5. Login to ABC_PetroCorp(PXXXXXX) IDP with Emily's credentials Emily/Test@12345</p>	 <p>The screenshot shows the 'Log On' page of the SAP Cloud Platform Portal. It features input fields for 'E-Mail' and 'Password', a 'Remember me' checkbox, and a prominent blue 'Log On' button. Below the button is a link to 'Forgot password?'</p>
<p>6. This will open the SAP Cloud Platform Portal welcome screen.</p>	 <p>The screenshot shows the SAP Cloud Platform Portal's welcome screen. It includes a left sidebar with links like Home, Site Directory, Services and Tools, and Usage Analytics. The main area displays a 'Welcome to SAP Cloud Platform Portal' message, a 'Create New Site' button, and a 'Cannot find your site in the Site Directory?' link. There are also 'Latest highlights' and 'Watch video' sections.</p>
<p>7. Navigate to Site Directory from the left-hand side panel of SAP Cloud Platform Portal. Click on Import Site to open the import site dialog.</p>	 <p>The screenshot shows the 'Site Directory' screen of the SAP Cloud Platform Portal. It features a sidebar with the same navigation options as the welcome screen. The main content area has a large plus sign icon and a 'Import Site' button with a red arrow pointing to it. A search bar is also visible at the top right.</p>

Explanation	Screenshot
<p>8. Click on Browse to select mtar file.</p> <p>9. Choose the SAP_exported_solution_<...>.mtar file from the PollutionMonitoring project under the folder FLP folder as shown in the screenshot</p>	 <p>The screenshot shows the 'Import Site' dialog box. It has a text input field labeled 'File to import' and a 'Browse...' button. At the bottom are 'Import' and 'Cancel' buttons. The text above the input field says 'File must be a .zip or .mtar file.'</p>
10. Click on the Import	 <p>The screenshot shows a file explorer window with the path 'C:\Users\student\cpl282\cloud-pollutionmonitoringapp-master\pollutionmonitoring\FLP'. The 'FLP' folder is highlighted with a red box. Inside the 'FLP' folder, there is a file named 'SAP_exported_solution_1c50e2f9-aed8-4e0b-a3ab-49bd5790649c.mtar', which is also highlighted with a red box.</p>

Explanation	Screenshot
<p>11. Provide SAP User ID: P1942783XXX Password: Welcome17 Email: cpl282- XXX@techd.cloud.sap</p> <p>to access SAP HANA Cloud Platform account and click on OK.</p>	

Step2: Publish and Launch SAP Fiori Launchpad site from ABC PetroCorp subaccount

Explanation	Screenshot
<p>1. Click on the Edit menu of the newly created tile for Plant Management site.</p> <p>2. Click on Publish.</p>	
<p>3. In the Publish Site confirmation dialog Click on Publish. This will publish the site and make it available to use.</p>	
<p>4. Click on the SAP FLP Plant Management Site URL. This will launch the SAP FLP site in a new browser tab.</p>	

Done!

With this, employees of **ABC PetroCorp** will be able to access the Pollution monitoring dashboard application using SAP Fiori Launchpad.

www.sap.com/contactsap

© 2017 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company.

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platform directions and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies. See http://www.sap.com/corporate-en/legal/copyright/index_enx for additional trademark information and notices.

