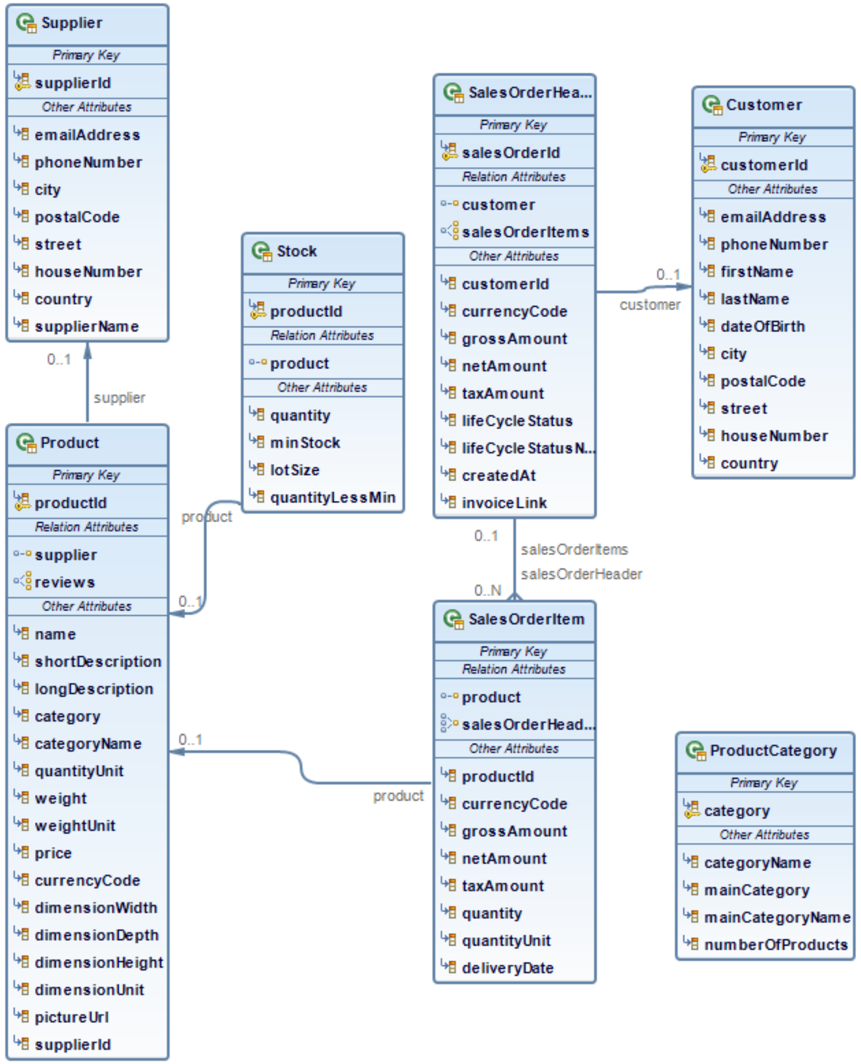
Week 2 Unit 2: **Developing the JPA Model – Practice**

Developing the JPA Model – Practice

Recap: The JPA model of the ESPM application



Supplier
Primary Key
- supplierId

Other Attributes
- emailAddress
- phoneNumber
- city
- postalCode
- street
- houseNumber
- country
- supplierName

Stock
Primary Key
- productId

Relation Attributes
- product

Other Attributes
- quantity
- minStock
- lotSize
- quantityLessMin

SalesOrderHea...
Primary Key
- salesOrderId

Relation Attributes
- customer
- salesOrderItems

Other Attributes
- customerId
- currencyCode
- grossAmount
- netAmount
- taxAmount
- lifeCycleStatus
- lifeCycleStatusN...
- createdAt
- invoiceLink

Customer
Primary Key
- customerId

Other Attributes
- emailAddress
- phoneNumber
- firstName
- lastName
- dateOfBirth
- city
- postalCode
- street
- houseNumber
- country

Product
Primary Key
- productId

Relation Attributes
- supplier
- reviews

Other Attributes
- name
- shortDescription
- longDescription
- category
- categoryName
- quantityUnit
- weight
- weightUnit
- price
- currencyCode
- dimensionWidth
- dimensionDepth
- dimensionHeight
- dimensionUnit
- pictureUrl
- supplierId

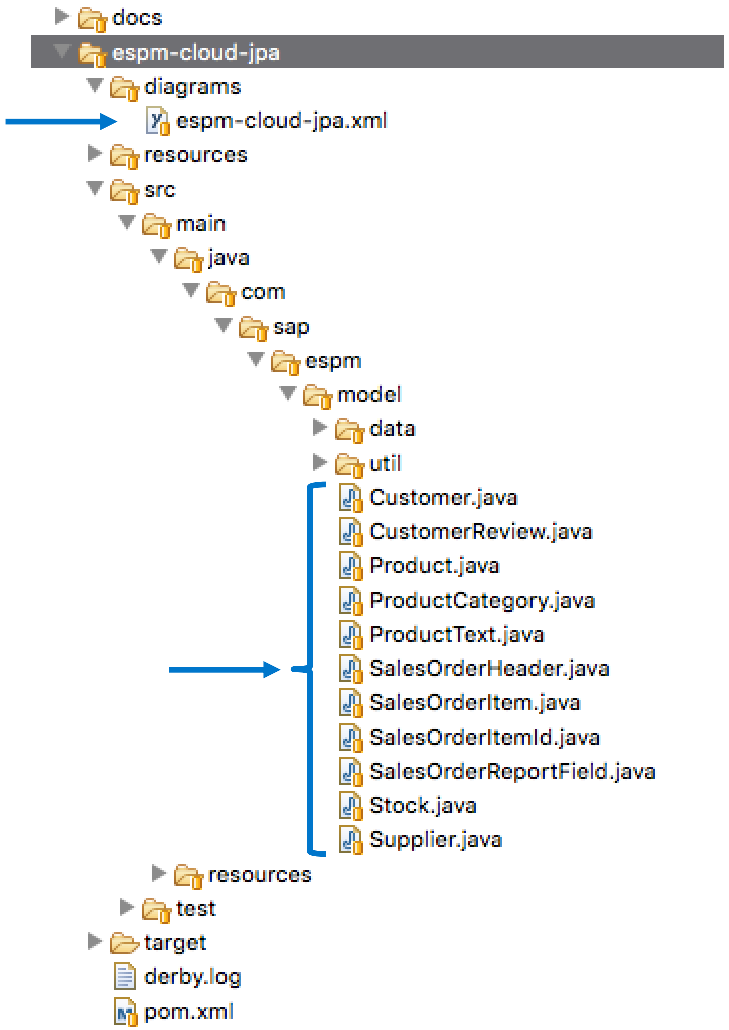
SalesOrderItem
Primary Key

Relation Attributes
- product
- salesOrderHead...

Other Attributes
- productId
- currencyCode
- grossAmount
- netAmount
- taxAmount
- quantity
- quantityUnit
- deliveryDate

ProductCategory
Primary Key
- category

Other Attributes
- categoryName
- mainCategory
- mainCategoryName
- numberOfProducts



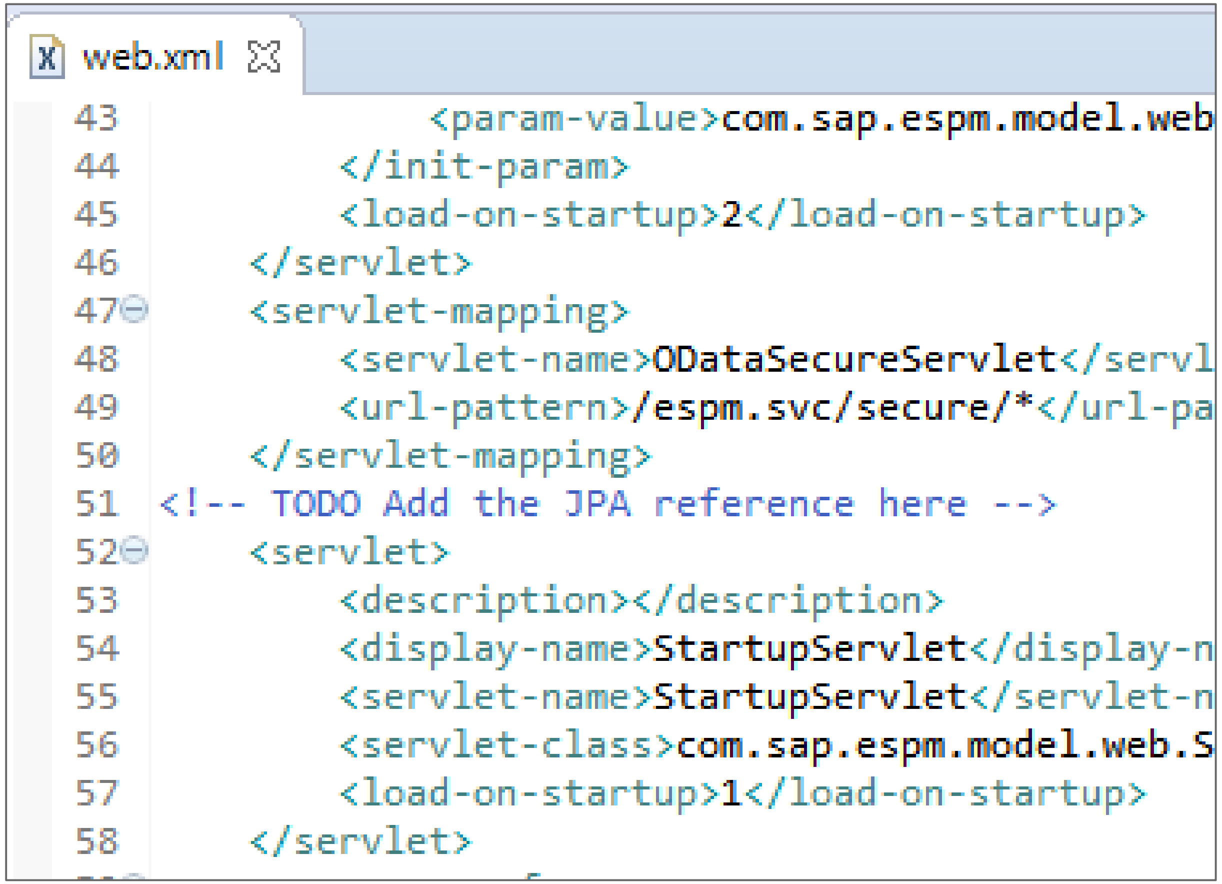
```
▶ docs
▼ espm-cloud-jpa
  ▼ diagrams
      espm-cloud-jpa.xml
  ▶ resources
  ▼ src
    ▼ main
      ▼ java
        ▼ com
          ▼ sap
            ▼ espm
              ▼ model
                ▶ data
                ▶ util
                  Customer.java
                  CustomerReview.java
                  Product.java
                  ProductCategory.java
                  ProductText.java
                  SalesOrderHeader.java
                  SalesOrderItem.java
                  SalesOrderItemId.java
                  SalesOrderReportField.java
                  Stock.java
                  Supplier.java
      ▶ resources
    ▶ test
  ▶ target
    derby.log
    pom.xml
```

Developing the JPA Model – Practice

Hands-on (1)

Prepare the Web Application Project for JPA

- In Eclipse, navigate to the project espm-cloud-web
- Navigate to src/main/webapp/WEB-INF
- Open the web.xml file
- Replace the to-do part as shown in the screenshot with code snippet 1 from the file openSAP_hcp2_Week_2_Unit_2_Exercise
- Save your changes



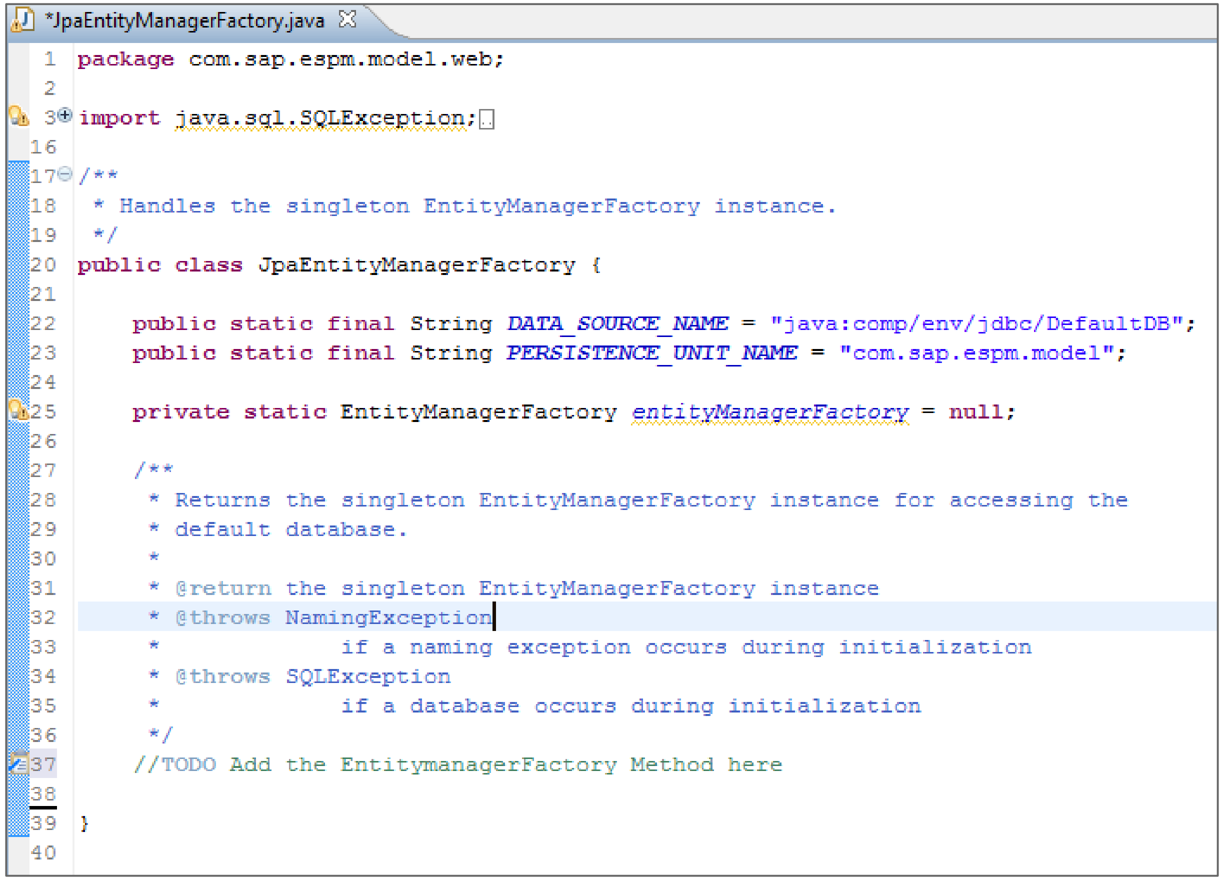
```
            <param-value>com.sap.espm.model.web
        </init-param>
        <load-on-startup>2</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>ODataSecureServlet</servl
        <url-pattern>/espm.svc/secure/*</url-pa
    </servlet-mapping>
<!-- TODO Add the JPA reference here -->
    <servlet>
        <description></description>
        <display-name>StartupServlet</display-n
        <servlet-name>StartupServlet</servlet-n
        <servlet-class>com.sap.espm.model.web.S
        <load-on-startup>1</load-on-startup>
    </servlet>
```

Developing the JPA Model – Practice

Hands-on (2)

Initialize the EntityManagerFactory

- In Eclipse, navigate to the project espm-cloud-web
- Navigate to src/main/java → com.sap.espm.model.web
- Open the JpaEntityManagerFactory.java file
- Replace the to-do part as shown in the screenshot with code snippet 2 from the file openSAP_hcp2_Week_2_Unit_2_Exercise
- Save your changes



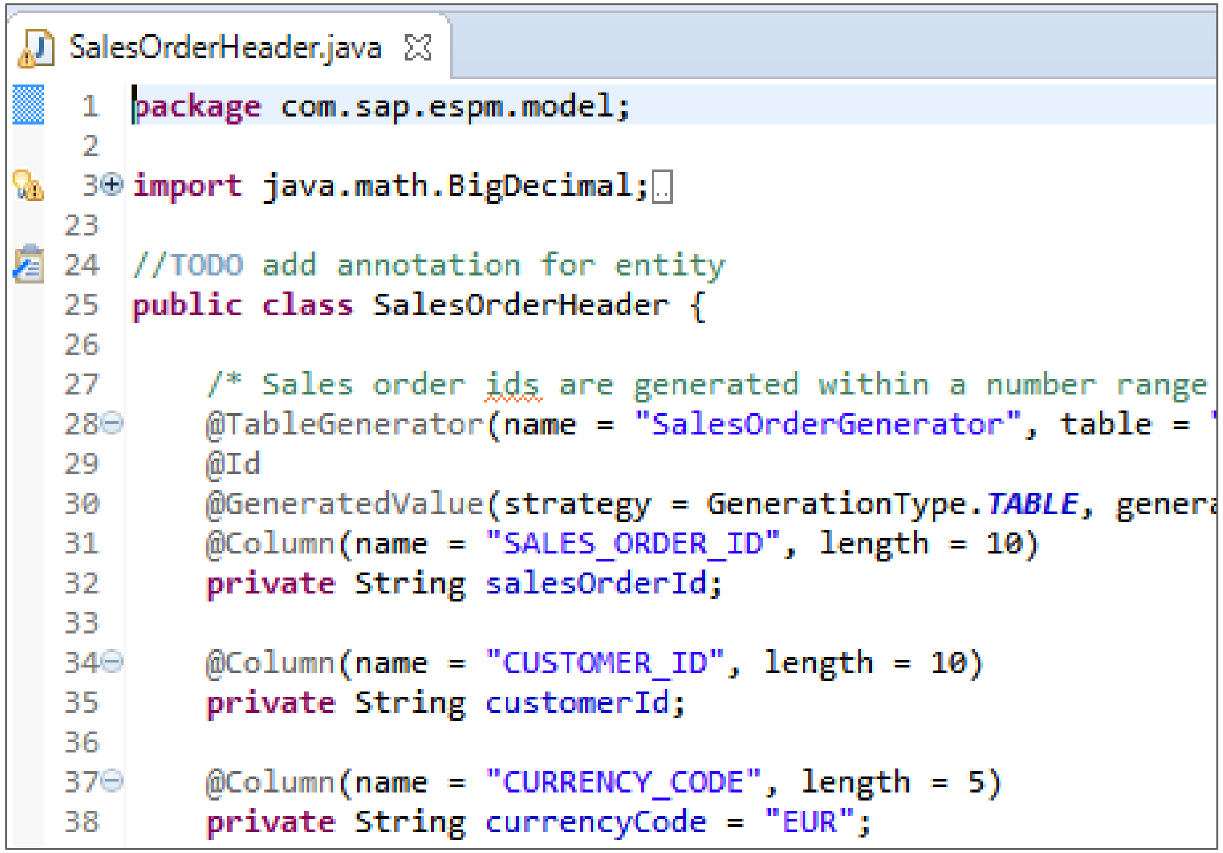
```
package com.sap.espm.model.web;

import java.sql.SQLException;

/**
 * Handles the singleton EntityManagerFactory instance.
 */
public class JpaEntityManagerFactory {

    public static final String DATA_SOURCE_NAME = "java:comp/env/jdbc/DefaultDB";
    public static final String PERSISTENCE_UNIT_NAME = "com.sap.espm.model";

    private static EntityManagerFactory entityManagerFactory = null;

    /**
     * Returns the singleton EntityManagerFactory instance for accessing the
     * default database.
     *
     * @return the singleton EntityManagerFactory instance
     * @throws NamingException
     *             if a naming exception occurs during initialization
     * @throws SQLException
     *             if a database occurs during initialization
     */
    //TODO Add the EntitymanagerFactory Method here

}
```

Developing the JPA Model – Practice

Hands-on (3)

Create an Entity – Part 1

- In Eclipse, navigate to the project espm-cloud-jpa
- Navigate to src/main/java → com.sap.espm.model
- Open the SalesOrderHeader.java file
- Replace the to-do part as shown in the screenshot with code snippet 3 from the file openSAP_hcp2_Week_2_Unit_2_Exercise
- Save your changes



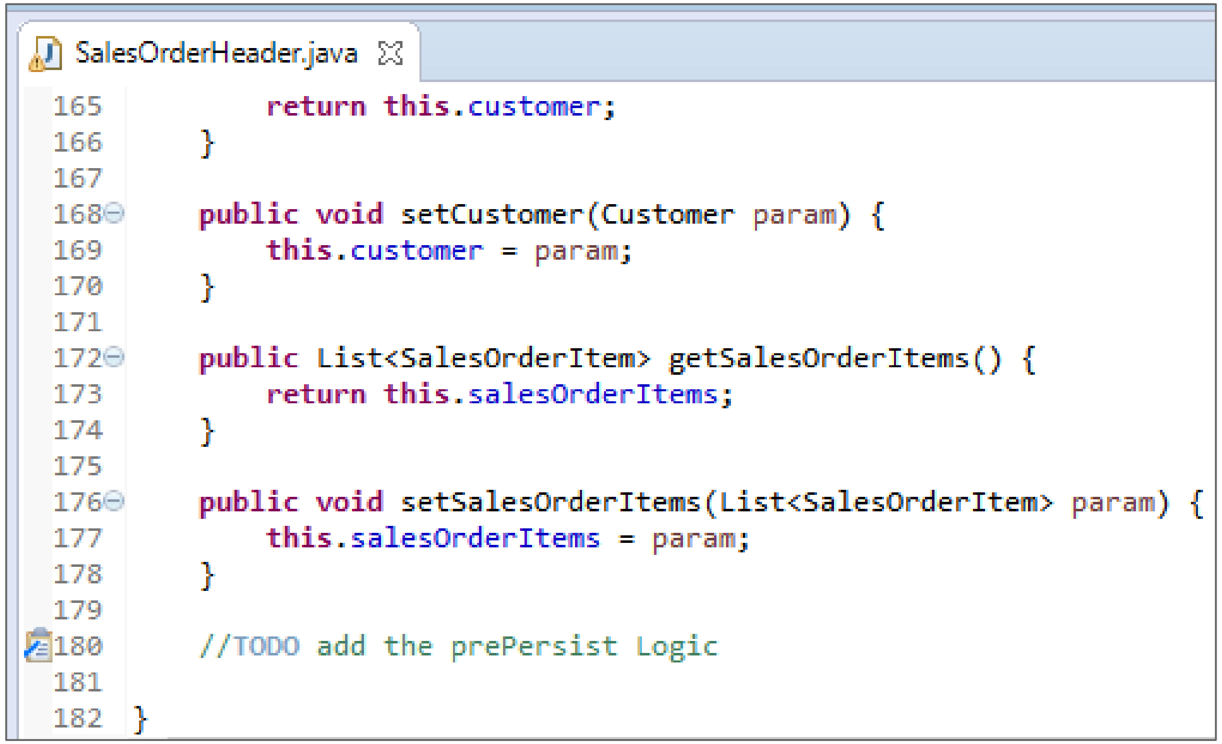
```
package com.sap.espm.model;

import java.math.BigDecimal;

//TODO add annotation for entity
public class SalesOrderHeader {

    /* Sales order ids are generated within a number range
    @TableGenerator(name = "SalesOrderGenerator", table = "
    @Id
    @GeneratedValue(strategy = GenerationType.TABLE, genera
    @Column(name = "SALES_ORDER_ID", length = 10)
    private String salesOrderId;

    @Column(name = "CUSTOMER_ID", length = 10)
    private String customerId;

    @Column(name = "CURRENCY_CODE", length = 5)
    private String currencyCode = "EUR";
```

Developing the JPA Model – Practice

Hands-on (4)

Create an Entity – Part 2

- In Eclipse, navigate to the project espm-cloud-jpa
- Navigate to src/main/java → com.sap.espm.model
- Open the SalesOrderHeader.java file
- Replace the to-do part as shown in the screenshot with code snippet 4 from the file openSAP_hcp2_Week_2_Unit_2_Exercise
- Save your changes



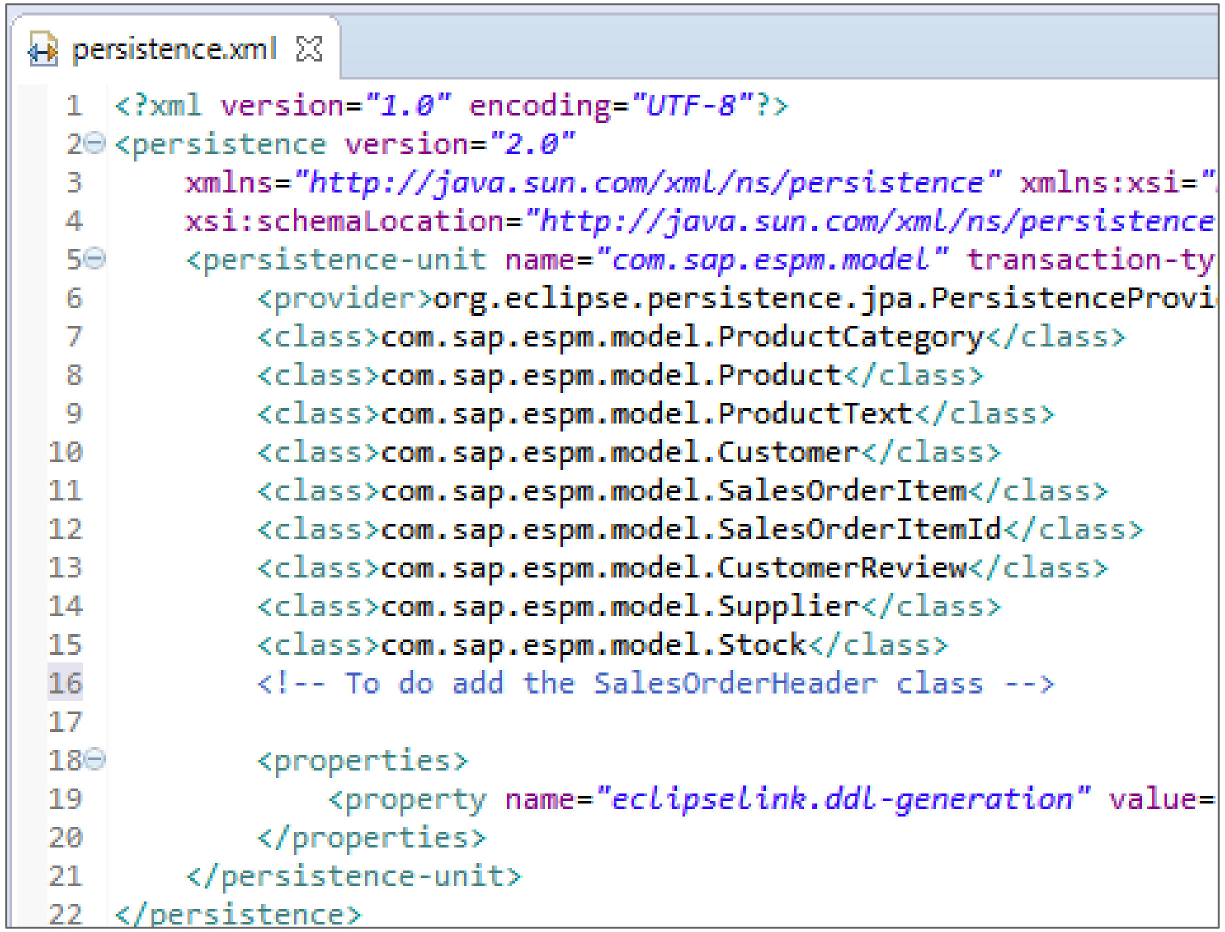
```
        return this.customer;
    }

    public void setCustomer(Customer param) {
        this.customer = param;
    }

    public List<SalesOrderItem> getSalesOrderItems() {
        return this.salesOrderItems;
    }

    public void setSalesOrderItems(List<SalesOrderItem> param) {
        this.salesOrderItems = param;
    }

    //TODO add the prePersist Logic

}
```

Developing the JPA Model – Practice

Hands-on (5)

Add class in persistence.xml

- In the Project Explorer view, expand the espm-cloud-jpa/JPA-Content
- Open the persistence.xml file
- Replace the to-do part as shown in the screenshot with code snippet 5 from the file openSAP_hcp2_Week_2_Unit_2_Exercise
- Save the persistence.xml
- After building the project you can run it



```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.0"
    xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="
    xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
    <persistence-unit name="com.sap.espm.model" transaction-ty
        <provider>org.eclipse.persistence.jpa.PersistenceProvi
        <class>com.sap.espm.model.ProductCategory</class>
        <class>com.sap.espm.model.Product</class>
        <class>com.sap.espm.model.ProductText</class>
        <class>com.sap.espm.model.Customer</class>
        <class>com.sap.espm.model.SalesOrderItem</class>
        <class>com.sap.espm.model.SalesOrderItemId</class>
        <class>com.sap.espm.model.CustomerReview</class>
        <class>com.sap.espm.model.Supplier</class>
        <class>com.sap.espm.model.Stock</class>
        <!-- To do add the SalesOrderHeader class -->

        <properties>
            <property name="eclipselink.ddl-generation" value=
        </properties>
    </persistence-unit>
</persistence>
```

Developing the JPA Model – Practice

What you’ve learned in this unit

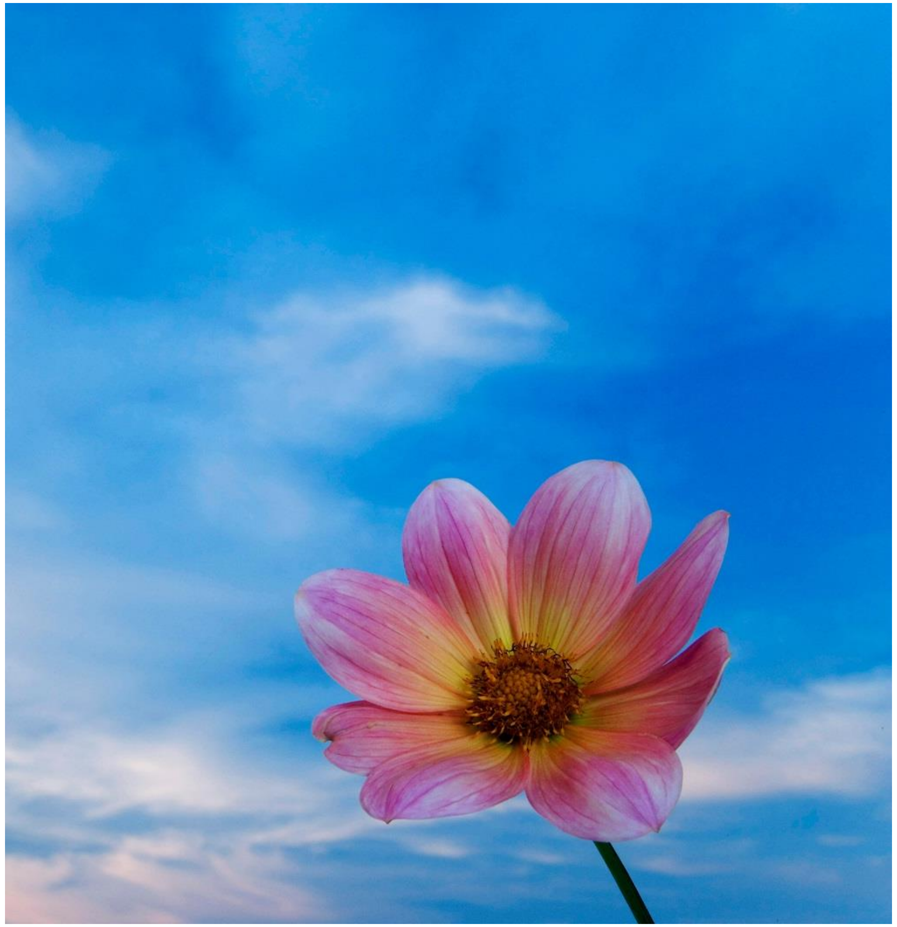
- Recap: The JPA model of the ESPM application
- How to initialize the EntityManagerFactory
- How to use JPA entity annotations
- How to edit the JPA persistence.xml file

Developing the JPA Model – Practice

Further reading

i Additional Material

- Persistence Service in SAP HANA Cloud Platform – https://help.hana.ondemand.com/help/frameset.htm?e7b3c275bb571014a910b3fb4329cf09.html



Thank you

Contact information:

open@sap.com

openSAP

© 2016 SAP SE or an SAP affiliate company. All rights reserved.

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 SE or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP SE 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, which speak only as of their dates, and they should not be relied upon in making purchasing decisions.