MODULE 2

Building the Baseline

Liferay

Building the Baseline

Learning Objectives

- Understand when it makes sense to develop a custom application for Liferay DXP
- Learn about Service Builder and service.xml

Tasks to Accomplish

- Create the Assignment Service
- Implement Assignment Local Service
- Implement Assignment Remote Service

Exercise Prerequisites

- Java JDK installed to run Liferay
  - Download here: https://www.oracle.com/technetwork/java/javase/downloads/jdk11-downloads-5066655.html
  - Instructions on installation here: https://www.java.com/en/download/help/download_options.xml
- Liferay Developer Studio installed with the "Gradebook Workspace" already created
  - This was done in the previous training module

Table of Contents

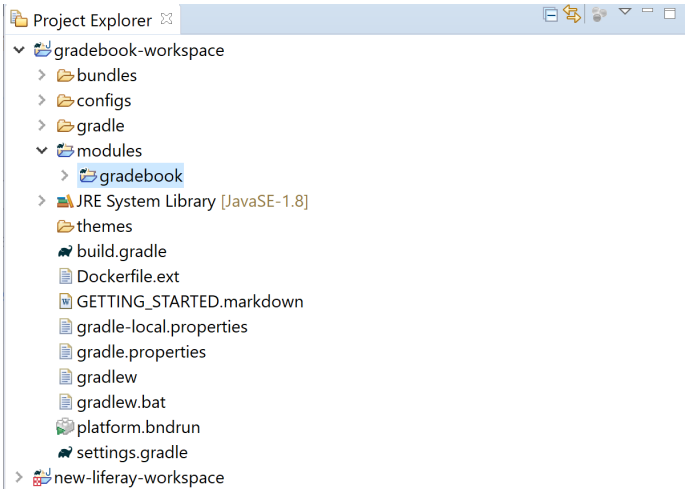
Create the Assignment Service

Exercise Goals

- Create a Liferay Service Builder Project using the *service-builder* template
- Define the Assignment entity
- Define service exceptions
- Final code review
- Build the service

Create a Liferay Service Builder Project

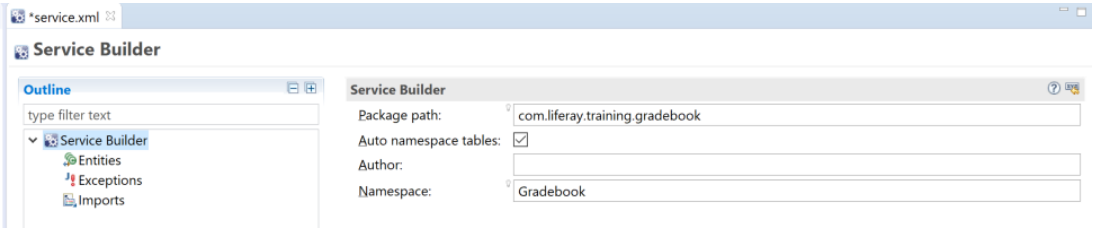
1. **Launch** Developer Studio if it's not already running.
    - Make sure you are using the gradebook-workspace
2. **Create** a new *Liferay Module Project* in the gradebook-workspace.
3. **Type** `gradebook` for *Project Name*.
4. **Choose** *service-builder* for *Project Template Name*.
5. **Click** *Next*.
6. **Type** `com.liferay.training.gradebook` for *Package name*.
7. **Click** *Finish*.



`service.xml` is the main configuration file of a Service Builder project. It lets you define model entities, data sources, finder methods, and exceptions for your service. You can customize `service.xml` with a graphical designer tool or edit the file's source code directly.

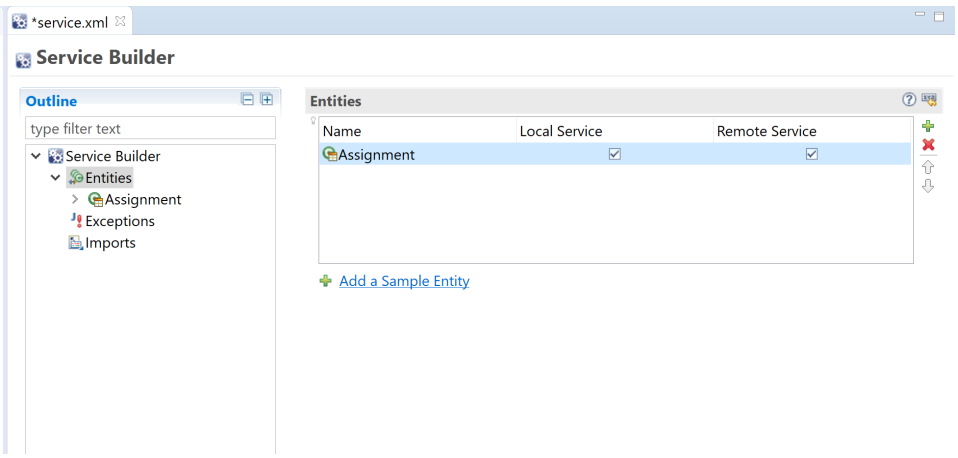
Define the Assignment Entity

1. **Go to** `modules > gradebook > gradebook-service` in the *Project Explorer*.
2. **Open** the `service.xml` file.
3. **Click** on the *Overview* view.
4. **Click** *Service Builder* in the outline tree.
5. **Enter** "Gradebook" in the *Namespace*.
6. **Right-click** on the `Foo` entity in the outline tree
7. **Click** *Delete*.



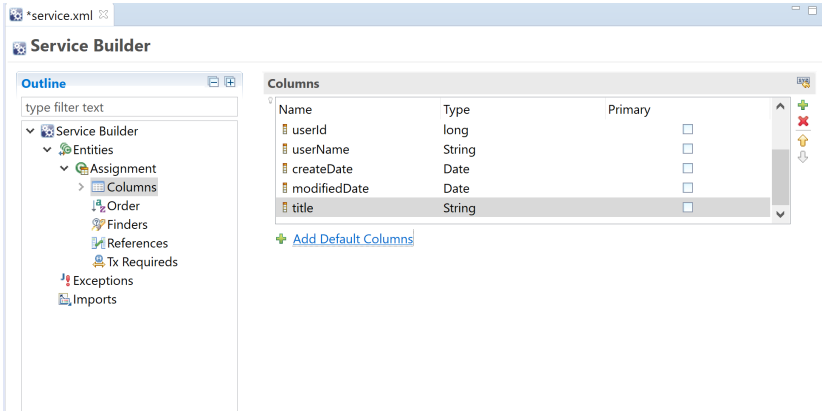
Create the Assignment Entity

1. **Click** the green plus icon on the right side of the entities list to add a new entity.
2. **Type** `Assignment` for the *Name* field.
3. **Check** both the *Local Service* and the *Remote Service*.



Define Assignment Columns

1. **Double-click** on the *Assignment* entity in the outline tree to open entity properties.
2. **Click** *Columns*.
3. **Click** *Add Default Columns* to add a default set of fields.
4. **Click** the green plus sign on the right side of the columns list to add a new column.
5. **Type** `title` for *Name*.
6. **Double-click** the *Type* field for the title column.
7. **Click** the browse icon on the right side of the field.
8. **Choose** *String*.



Edit the Column Definitions for the Assignment Entity

1. **Click** on the *Source* tab in the designer.

2. **Add** the rest of the Assignment's columns after the *title* column:

```
<column name="description" type="String"></column>

<column name="dueDate" type="Date"></column>
```

```
<!-- Audit fields -->

<column name="companyId" type="Long"></column>
<column name="userId" type="Long"></column>
<column name="userName" type="String"></column>
<column name="createDate" type="Date"></column>
<column name="modifiedDate" type="Date"></column>
<column name="title" type="String"></column>
<column name="description" type="String"></column>
<column name="dueDate" type="Date"></column>
```

Add Definitions to the `service.xml` File

1. **Add** the following snippet after the *column* definitions:

```
<!-- Order -->
<order by="asc">
        <order-column name="title" />
</order>
```

2. **Add** the following snippet after the *order* definition:

```
<!-- Finders -->
<!-- Find by groupId -->
<finder name="GroupId" return-type="Collection">
        <finder-column name="groupId"></finder-column>
</finder>
```

> NOTE:
> References define entity services injected in our service classes. This helps to keep the database calls inside a single transaction. We need the Group services and Liferay Asset services for integrating to the Liferay Asset framework for later exercise steps.

3. **Add** the following reference definitions after the *finder* definitions:

```
<!-- Reference to Group entity service -->
<reference entity="Group" package-path="com.liferay.portal"></reference>
<!-- Entity services needed for the integration to Asset framework -->
<reference entity="AssetEntry"
        package-path="com.liferay.portlet.asset"></reference>
<reference entity="AssetLink"
        package-path="com.liferay.portlet.asset"></reference>
<reference entity="AssetTag"
        package-path="com.liferay.portlet.asset"></reference>
```

4. **Add** the following code snippet after the closing tag of *entity*:

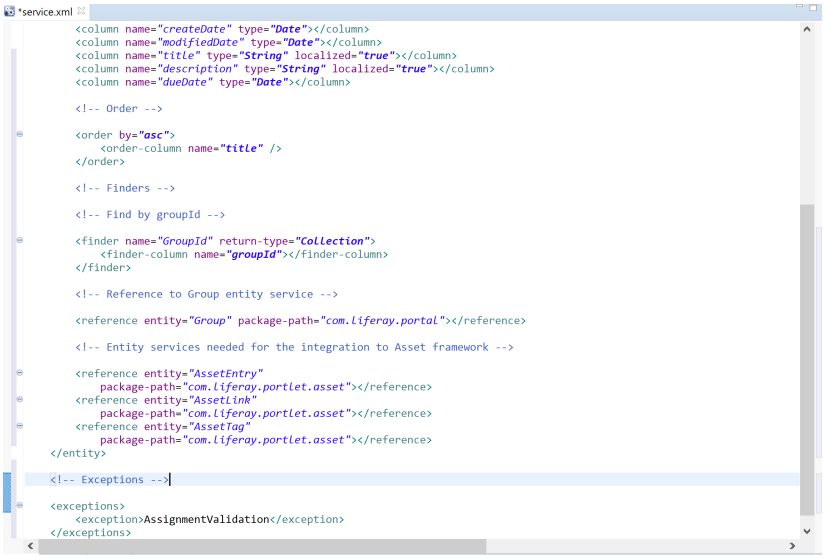
```
<!-- Exceptions -->

<exceptions>

        <exception>AssignmentValidation</exception>

</exceptions>
```

5. **Save** the *service.xml* file.



Use Liferay Developer Studio's automatic code formatting to fix indents and spacing. The final `service.xml` should look like this:

Final Code Review

```
<?xml version="1.0"?>
<!DOCTYPE service-builder PUBLIC "-//Liferay//DTD Service Builder 7.3.0//EN" "http://www.

<service-builder dependency-injector="ds" package-path="com.liferay.training.gradebook">
        <namespace>Gradebook</namespace>
        <!--<entity data-source="sampleDataSource" local-service="true" name="Foo" remote
        <entity name="Assignment" local-service="true">

                <!-- PK fields -->
```

```
<column name="assignmentId" primary="true" type="long"></column>

<!-- Group instance -->

<column name="groupId" type="long"></column>

<!-- Audit fields -->

<column name="companyId" type="long"></column>
<column name="userId" type="long"></column>
<column name="userName" type="String"></column>
<column name="createDate" type="Date"></column>
<column name="modifiedDate" type="Date"></column>
<column name="title" type="String"></column>
<column name="description" type="String"></column>
<column name="dueDate" type="Date"></column>

<!-- Order -->

<order by="asc">
        <order-column name="title" />
</order>

<!-- Finders -->

<!-- Find by groupId -->

<finder name="GroupId" return-type="Collection">
        <finder-column name="groupId"></finder-column>
</finder>
```

```
            <!-- Reference to Group entity service -->

            <reference entity="Group" package-path="com.liferay.portal"></reference>

            <!-- Entity services needed for the integration to Asset framework -->

            <reference entity="AssetEntry"
                    package-path="com.liferay.portlet.asset"></reference>
            <reference entity="AssetLink"
                    package-path="com.liferay.portlet.asset"></reference>
            <reference entity="AssetTag"
                    package-path="com.liferay.portlet.asset"></reference>
        </entity>

        <!-- Exceptions -->

        <exceptions>
            <exception>AssignmentValidation</exception>
        </exceptions>
</service-builder>
```

When you run the `buildService` Gradle task, the following items are generated:

- Database schema for the service (committed at module deploy time)
- Persistence and caching
- Local and remote service APIs

> NOTE:
>
> Remember that you have to rebuild services whenever you edit the `service.xml`.

Build the Service

1. **Expand** the `gradebook-workspace/modules/gradebook/build` in the *Gradle Tasks* panel.
2. **Run** the `buildService` task.
    - Note that you can run the service generation task several different ways:
        - The *Gradle Tasks* panel using the project's `buildService` task
        - The *Overview* panel of the `service.xml` designer
        - The *Context* menu of the gradebook-service project
        - The *Command Line* using the Liferay Workspace `gradlew` script
3. **Configure** and *save* the dependencies in the gradebook-service module's `build.gradle` file as follows:

```
dependencies {
        compileOnly group: "com.liferay.portal", name: "release.dxp.api"
        compileOnly project(":modules:gradebook:gradebook-api")
}

buildService {
        apiDir = "../gradebook-api/src/main/java"
}

group = "com.liferay.training.gradebook"

tasks.withType(JavaCompile) {

        // Generated classes using Jodd library are unable to be read when compiled agair

        sourceCompatibility = JavaVersion.VERSION_1_8
        targetCompatibility = JavaVersion.VERSION_1_8
}
```

```
dependencies {
    compileOnly group: "com.liferay.portal", name: "release.dxp.api"
    compileOnly project(":modules:gradebook:gradebook-api")
}

buildService {
    apiDir = "../gradebook-api/src/main/java"
}

group = "com.liferay.training.gradebook"

tasks.withType(JavaCompile) {

    // Generated classes using Jodd library are unable to be read when compiled against JDK 11

    sourceCompatibility = JavaVersion.VERSION_1_8
    targetCompatibility = JavaVersion.VERSION_1_8
}
```

Implement Assignment Local Service

Exercise Goals

- Implement `addAssignment()`
- Implement `updateAssignment()`
- Implement the finder methods
- Silence generated methods
- Do a final code review
- Rebuild the service

Before implementing the method for adding assignments, open the local service base class `AssignmentLocalServiceBaseImpl` and take a look at the generated `addAssigment()` method. This method doesn't automatically generate an ID, set the audit fields (like creation or modification date), or validate the entity. Creating an overload for `addAssignment()` will take care of these tasks.

```
@Indexable(type = IndexableType.REINDEX)
@Override
public Assignment addAssignment(Assignment assignment) {
        assignment.setNew(true);

        return assignmentPersistence.update(assignment);
}
```

Implement addAssignment()

1. **Open** the `AssignmentLocalServiceImpl`. The empty class looks like this:

```
/**
 * Copyright (c) 2000-present Liferay, Inc. All rights reserved.
 *
```

```
 * This library is free software; you can redistribute it and/or modify it u
 * the terms of the GNU Lesser General Public License as published by the Fr
 * Software Foundation; either version 2.1 of the License, or (at your optio
 * any later version.
 *
 * This library is distributed in the hope that it will be useful, but WITHO
 * ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FIT
 * FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for m
 * details.
 */
package com.liferay.training.gradebook.service.impl;
import com.liferay.portal.aop.AopService;
import com.liferay.training.gradebook.service.base.AssignmentLocalServiceBas
import org.osgi.service.component.annotations.Component;
/**
 * The implementation of the assignment local service.
 *
 * <p>
 * All custom service methods should be put in this class. Whenever methods
 *
 * <p>
 * This is a local service. Methods of this service will not have security c
 * </p>
 *
 * @author Brian Wing Shun Chan
 * @see AssignmentLocalServiceBaseImpl
 */
@Component(
        property = "model.class.name=com.liferay.training.gradebook.model.As
        service = AopService.class
)
public class AssignmentLocalServiceImpl extends AssignmentLocalServiceBaseIm
```

```
        /*
         * NOTE FOR DEVELOPERS:
         *
         * Never reference this class directly. Use <code>com.liferay.train
         */
    }
```

2. **Implement** the `addAssignment()` in the class as follows:

```
public class AssignmentLocalServiceImpl extends AssignmentLocalServiceBaseImpl {

   public Assignment addAssignment(long groupId, String title, String description,
         Date dueDate, ServiceContext serviceContext) throws PortalException {
  // Validate assignment parameters.
      _assignmentValidator.validate(title, description, dueDate);
      // Get group and user.
      Group group = groupLocalService.getGroup(groupId);
      long userId = serviceContext.getUserId();
      User user = userLocalService.getUser(userId);
      // Generate primary key for the assignment.
      long assignmentId = counterLocalService.increment(Assignment.class.getName());
      // Create assigment. This doesn't yet persist the entity.
      Assignment assignment = createAssignment(assignmentId);
      // Populate fields.
      assignment.setCompanyId(group.getCompanyId());
      assignment.setCreateDate(serviceContext.getCreateDate(new Date()));
      assignment.setDueDate(dueDate);
      assignment.setDescription(description);
      assignment.setGroupId(groupId);
      assignment.setModifiedDate(serviceContext.getModifiedDate(new Date()));
      assignment.setTitle(title);
      assignment.setUserId(userId);
      assignment.setUserName(user.getScreenName());

  // Persist assignment to database.
  return super.addAssignment(assignment);
}
```

Implement updateAssignment

1. **Create** an overload for the `updateAssignment()`:

```
public Assignment updateAssignment(
        long assignmentId, Map<Locale, String> titleMap, Map<Locale, String> descriptionM
        Date dueDate, ServiceContext serviceContext)
        throws PortalException {
        // Get the Assignment by id.
        Assignment assignment = getAssignment(assignmentId);
        // Set updated fields and modification date.
        assignment.setModifiedDate(new Date());
        assignment.setTitle(title);
        assignment.setDueDate(dueDate);
        assignment.setDescription(description);
        assignment = super.updateAssignment(assignment);
        return assignment;
}
```

Defining finders in `service.xml` automatically creates the corresponding methods in the persistence classes, but we cannot access those directly from the controller layer and have to implement facades in the service implementation class.

Implement the Finder Methods

1. **Implement** the finder methods as follows:

```
public List<Assignment> getAssignmentsByGroupId(long groupId) {
  return assignmentPersistence.findByGroupId(groupId);
}
public List<Assignment> getAssignmentsByGroupId(long groupId, int start, int end) {
  return assignmentPersistence.findByGroupId(groupId, start, end);
}
public List<Assignment> getAssignmentsByGroupId(long groupId, int start, int end,
     OrderByComparator<Assignment> orderByComparator) {
  return assignmentPersistence.findByGroupId(groupId, start, end, orderByComparator);
}
```

```
public List<Assignment> getAssignmentsByKeywords(
  long groupId, String keywords, int start, int end,
  OrderByComparator<Assignment> orderByComparator) {
  return assignmentLocalService.dynamicQuery(
     getKeywordSearchDynamicQuery(groupId, keywords), start, end,
     orderByComparator);
}
public long getAssignmentsCountByKeywords(long groupId, String keywords) {
  return assignmentLocalService.dynamicQueryCount(
     getKeywordSearchDynamicQuery(groupId, keywords));
}
private DynamicQuery getKeywordSearchDynamicQuery(
  long groupId, String keywords) {
  DynamicQuery dynamicQuery = dynamicQuery().add(
     RestrictionsFactoryUtil.eq("groupId", groupId));
  if (Validator.isNotNull(keywords)) {
     Disjunction disjunctionQuery =
        RestrictionsFactoryUtil.disjunction();
     disjunctionQuery.add(
        RestrictionsFactoryUtil.like("title", "%" + keywords + "%"));
     disjunctionQuery.add(
        RestrictionsFactoryUtil.like(
           "description", "%" + keywords + "%"));
     dynamicQuery.add(disjunctionQuery);
  }
  return dynamicQuery;
}
```

> NOTE:
>
> For the sake of this exercise, we introduced a custom `getAssignmentsByKeywords()` method here, which we will use on the user interface later for searching. This method is using Dynamic Queries, which allow you to query the database with custom SQL. Note that this specific query wouldn't work well with localized fields, which are stored in xml.

Sometimes it's practical to silence generated methods to ensure correct API usage. Override and "silence" the generated `addAssignment()` and `updateAssignment()` method signatures, which we replaced with our overrides before.

"Silence" the Generated Method

1. Add the following code to the end of the `AssignmentLocalServiceImpl.java` class.

```
 @Override
public Assignment addAssignment(Assignment assignment) {
  throw new UnsupportedOperationException("Not supported.");
}
@Override
public Assignment updateAssignment(Assignment assignment) {
  throw new UnsupportedOperationException("Not supported.");
}
```

Do a Final Code Review

1. **Resolve** missing imports.
2. **Fix** indents and spacing by using automatic code formatting.
3. **Save** the file.
    - The final `AssignmentLocalServiceImpl.java` class will look like this:

```
/**
 * Copyright (c) 2000-present Liferay, Inc. All rights reserved.
```

```
 *
 * This library is free software; you can redistribute it and/or modify it under
 * the terms of the GNU Lesser General Public License as published by the Free
 * Software Foundation; either version 2.1 of the License, or (at your option)
 * any later version.
 *
 * This library is distributed in the hope that it will be useful, but WITHOUT
 * ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
 * FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more
 * details.
 */
package com.liferay.training.gradebook.service.impl;
import com.liferay.portal.aop.AopService;
import com.liferay.portal.kernel.dao.orm.Disjunction;
import com.liferay.portal.kernel.dao.orm.DynamicQuery;
import com.liferay.portal.kernel.dao.orm.RestrictionsFactoryUtil;
import com.liferay.portal.kernel.exception.PortalException;
import com.liferay.portal.kernel.model.Group;
import com.liferay.portal.kernel.model.User;
import com.liferay.portal.kernel.service.ServiceContext;
import com.liferay.portal.kernel.util.OrderByComparator;
import com.liferay.portal.kernel.util.Validator;
import com.liferay.training.gradebook.model.Assignment;
import com.liferay.training.gradebook.service.base.AssignmentLocalServiceBaseImpl;
import java.util.Date;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import org.osgi.service.component.annotations.Component;
/**
 * The implementation of the assignment local service.
 *
```

```
 * <p>
 * All custom service methods should be put in this class. Whenever methods are
 * added, rerun ServiceBuilder to copy their definitions into the
 * <code>com.liferay.training.gradebook.service.AssignmentLocalService</code>
 * interface.
 *
 * <p>
 * This is a local service. Methods of this service will not have security
 * checks based on the propagated JAAS credentials because this service can only
 * be accessed from within the same VM.
 * </p>
 *
 * @author Brian Wing Shun Chan
 * @see AssignmentLocalServiceBaseImpl
 */
@Component(
   property = "model.class.name=com.liferay.training.gradebook.model.Assignment",
   service = AopService.class
)
public class AssignmentLocalServiceImpl extends AssignmentLocalServiceBaseImpl {

   public Assignment addAssignment(long groupId, String title, String description,
         Date dueDate, ServiceContext serviceContext) throws PortalException {
      // Get group and user.
      Group group = groupLocalService.getGroup(groupId);
      long userId = serviceContext.getUserId();
      User user = userLocalService.getUser(userId);
      // Generate primary key for the assignment.
      long assignmentId = counterLocalService.increment(Assignment.class.getName());
      // Create assigment. This doesn't yet persist the entity.
      Assignment assignment = createAssignment(assignmentId);
      // Populate fields.
```

```
   assignment.setCompanyId(group.getCompanyId());
   assignment.setCreateDate(serviceContext.getCreateDate(new Date()));
   assignment.setDueDate(dueDate);
   assignment.setDescription(description);
   assignment.setGroupId(groupId);
   assignment.setModifiedDate(serviceContext.getModifiedDate(new Date()));
   assignment.setTitle(title);
   assignment.setUserId(userId);
   assignment.setUserName(user.getScreenName());
   // Persist assignment to database.
   return super.addAssignment(assignment);
}

public Assignment updateAssignment(long assignmentId, String title,
      String description, Date dueDate, ServiceContext serviceContext) throws PortalE
   // Get the Assignment by id.
   Assignment assignment = getAssignment(assignmentId);
   // Set updated fields and modification date.
   assignment.setModifiedDate(new Date());
   assignment.setTitle(title);
   assignment.setDueDate(dueDate);
   assignment.setDescription(description);
   assignment = super.updateAssignment(assignment);
   return assignment;
}

public List<Assignment> getAssignmentsByGroupId(long groupId) {
   return assignmentPersistence.findByGroupId(groupId);
}
public List<Assignment> getAssignmentsByGroupId(long groupId, int start, int end) {
   return assignmentPersistence.findByGroupId(groupId, start, end);
}
```

```
public List<Assignment> getAssignmentsByGroupId(long groupId, int start, int end,
      OrderByComparator<Assignment> orderByComparator) {
   return assignmentPersistence.findByGroupId(groupId, start, end, orderByComparator);
}
public List<Assignment> getAssignmentsByKeywords(
   long groupId, String keywords, int start, int end,
   OrderByComparator<Assignment> orderByComparator) {
   return assignmentLocalService.dynamicQuery(
      getKeywordSearchDynamicQuery(groupId, keywords), start, end,
      orderByComparator);
}
public long getAssignmentsCountByKeywords(long groupId, String keywords) {
   return assignmentLocalService.dynamicQueryCount(
      getKeywordSearchDynamicQuery(groupId, keywords));
}
private DynamicQuery getKeywordSearchDynamicQuery(
   long groupId, String keywords) {
   DynamicQuery dynamicQuery = dynamicQuery().add(
      RestrictionsFactoryUtil.eq("groupId", groupId));
   if (Validator.isNotNull(keywords)) {
      Disjunction disjunctionQuery =
         RestrictionsFactoryUtil.disjunction();
      disjunctionQuery.add(
         RestrictionsFactoryUtil.like("title", "%" + keywords + "%"));
      disjunctionQuery.add(
         RestrictionsFactoryUtil.like(
            "description", "%" + keywords + "%"));
      dynamicQuery.add(disjunctionQuery);
   }
   return dynamicQuery;
}
```

```
    @Override
    public Assignment addAssignment(Assignment assignment) {
        throw new UnsupportedOperationException("Not supported.");
    }
    @Override
    public Assignment updateAssignment(Assignment assignment) {
        throw new UnsupportedOperationException("Not supported.");
    }
}
```

Rebuild the Service

1. **Run** the `buildService` Gradle task at the gradebook level to regenerate the service.

Implement Assignment Remote Service

Exercise Goals

- Declare dependencies
- Implement the facade methods for the local service in the `AssignmentServiceImpl.java`
- Do a final code review
- Rebuild and deploy the service

Implement the Façade Methods

1. **Open** the `AssignmentServiceImpl.java` class. The empty class looks like this:

```
/**
* Copyright (c) 2000-present Liferay, Inc. All rights reserved.
*
* This library is free software; you can redistribute it and/or modify it under
* the terms of the GNU Lesser General Public License as published by the Free
* Software Foundation; either version 2.1 of the License, or (at your option)
* any later version.
*
* This library is distributed in the hope that it will be useful, but WITHOUT
* ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
* FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more
* details.
*/

package com.liferay.training.gradebook.service.impl;

import com.liferay.portal.aop.AopService;
```

```
import com.liferay.training.gradebook.service.base.AssignmentServiceBaseImpl;

import org.osgi.service.component.annotations.Component;

/**
* The implementation of the assignment remote service
*
* <p>
* All custom service methods should be put in this class. Whenever methods are adde
*
* <p>
* This is a remote service. Methods of this service are expected to have security c
* </p>
*
* @author Brian Wing Shun Chan
* @see AssignmentServiceBaseImpl
*/
@Component(
        property = {
                "json.web.service.context.name=gradebook",
                "json.web.service.context.path=Assignment"
        },
        service = AopService.class
)
public class AssignmentServiceImpl extends AssignmentServiceBaseImpl {

        /*
         * NOTE FOR DEVELOPERS:
         *
         * Never reference this class directly. Always use <code>com.liferay.traini
         */
}
```

2. **Implement** the façade methods in the class as follows:

```
public Assignment addAssignment(
 long groupId, String title, String description,
 Date dueDate, ServiceContext serviceContext)
 throws PortalException {
 return assignmentLocalService.addAssignment(
    groupId, title, description, dueDate, serviceContext);
}
public Assignment deleteAssignment(long assignmentId)
 throws PortalException {
 Assignment assignment =
    assignmentLocalService.getAssignment(assignmentId);
 return assignmentLocalService.deleteAssignment(assignment);
}
public Assignment getAssignment(long assignmentId)
 throws PortalException {
 Assignment assignment =
    assignmentLocalService.getAssignment(assignmentId);
 return assignment;
}
public List<Assignment> getAssignmentsByGroupId(long groupId) {
 return assignmentPersistence.findByGroupId(groupId);
}
public List<Assignment> getAssignmentsByKeywords(
 long groupId, String keywords, int start, int end,
 OrderByComparator<Assignment> orderByComparator) {
 return assignmentLocalService.getAssignmentsByKeywords(
    groupId, keywords, start, end, orderByComparator);
}
public long getAssignmentsCountByKeywords(long groupId, String keywords) {
```

```
        return assignmentLocalService.getAssignmentsCountByKeywords(
            groupId, keywords);
    }
    public Assignment updateAssignment(
        long assignmentId, String title, String description,
        Date dueDate, ServiceContext serviceContext)
        throws PortalException {
        return assignmentLocalService.updateAssignment(
            assignmentId, title, description, dueDate, serviceContext);
    }
```

Final Code Review

1. **Resolve** missing imports.
2. **Fix** indents and spacing by using automatic code formatting.
3. **Save** the file.

The complete file will look like this:

```
/**
 * Copyright (c) 2000-present Liferay, Inc. All rights reserved.
 *
 * This library is free software; you can redistribute it and/or modify it under
 * the terms of the GNU Lesser General Public License as published by the Free
 * Software Foundation; either version 2.1 of the License, or (at your option)
 * any later version.
 *
 * This library is distributed in the hope that it will be useful, but WITHOUT
 * ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
 * FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more
 * details.
 */
package com.liferay.training.gradebook.service.impl;
```

```
import com.liferay.portal.aop.AopService;

import com.liferay.portal.kernel.exception.PortalException;

import com.liferay.portal.kernel.service.ServiceContext;

import com.liferay.portal.kernel.util.OrderByComparator;

import com.liferay.training.gradebook.model.Assignment;

import com.liferay.training.gradebook.service.base.AssignmentServiceBaseImpl;

import java.util.Date;

import java.util.List;

import java.util.Locale;

import java.util.Map;

import org.osgi.service.component.annotations.Component;

/**

 * The implementation of the assignment remote service.

 *

 * <p>

 * All custom service methods should be put in this class. Whenever methods are added, re

 *

 * <p>

 * This is a remote service. Methods of this service are expected to have security checks

 * </p>

 *

 * @author Brian Wing Shun Chan

 * @see AssignmentServiceBaseImpl

 */

@Component(

   property = {

      "json.web.service.context.name=gradebook",

      "json.web.service.context.path=Assignment"

   },

   service = AopService.class

)

public class AssignmentServiceImpl extends AssignmentServiceBaseImpl {
```

```
public Assignment addAssignment(
   long groupId, String title, String description,
   Date dueDate, ServiceContext serviceContext)
   throws PortalException {
   return assignmentLocalService.addAssignment(
      groupId, title, description, dueDate, serviceContext);
}
public Assignment deleteAssignment(long assignmentId)
   throws PortalException {
   Assignment assignment =
      assignmentLocalService.getAssignment(assignmentId);
   return assignmentLocalService.deleteAssignment(assignment);
}
public Assignment getAssignment(long assignmentId)
   throws PortalException {
   Assignment assignment =
      assignmentLocalService.getAssignment(assignmentId);
   return assignment;
}
public List<Assignment> getAssignmentsByGroupId(long groupId) {
   return assignmentPersistence.findByGroupId(groupId);
}
public List<Assignment> getAssignmentsByKeywords(
   long groupId, String keywords, int start, int end,
   OrderByComparator<Assignment> orderByComparator) {
   return assignmentLocalService.getAssignmentsByKeywords(
      groupId, keywords, start, end, orderByComparator);
}
public long getAssignmentsCountByKeywords(long groupId, String keywords) {
   return assignmentLocalService.getAssignmentsCountByKeywords(
      groupId, keywords);
```

```
    }

    public Assignment updateAssignment(

        long assignmentId, String title, String description,

        Date dueDate, ServiceContext serviceContext)

        throws PortalException {

        return assignmentLocalService.updateAssignment(

            assignmentId, title, description, dueDate, serviceContext);

    }

}
```

Rebuild and Deploy the Service

1. **Run** the `buildService` Gradle task to regenerate the service.
2. **Start** the server if it's not running.
3. **Drag** the *gradebook-api* and *gradebook-service* modules onto the Liferay server to deploy the modules.

You should see a success log message if modules were deployed successfully:

```
2019-03-20 11:31:59.667 INFO  [pipe-start 984][BundleStartStopLogger:39] STARTED com.life

2019-03-20 11:32:02.573 INFO  [pipe-start 985][BundleStartStopLogger:39] STARTED com.life
```

Building the Baseline Module Quiz

1. Which of the following files does Service Builder use to generate code?

   A. serviceBuilder.xml

   B. service.xml

   C. ServiceBuilder.json

   D. serviceImpl.java

2. Which of the following are **not** ever generated by Service Builder? (choose two)

   A. Local

   B. Remote

   C. service.xml

   D. portlet-module-hints.xml

3. You should always rebuild your service after modifying the Local service.

   A. True

   B. False

4. The object that aggregates necessary information for features used throughout Liferay DXP's portlets and services is called?

   A. Service Context

   B. Portlet Context

   C. Request Parameters

   D. Permissions Aggregator

   E. Service Builder

5. Service Builder creates implementation classes for every defined entity's Model class and Entity Finders.

   A. True

   B. False

Answer Key

1. B
2. C, D
3. False
4. A
5. True