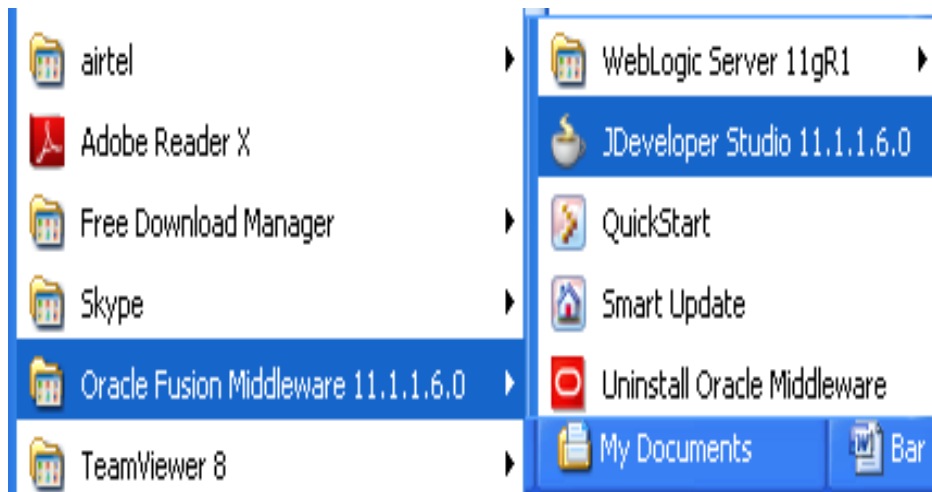


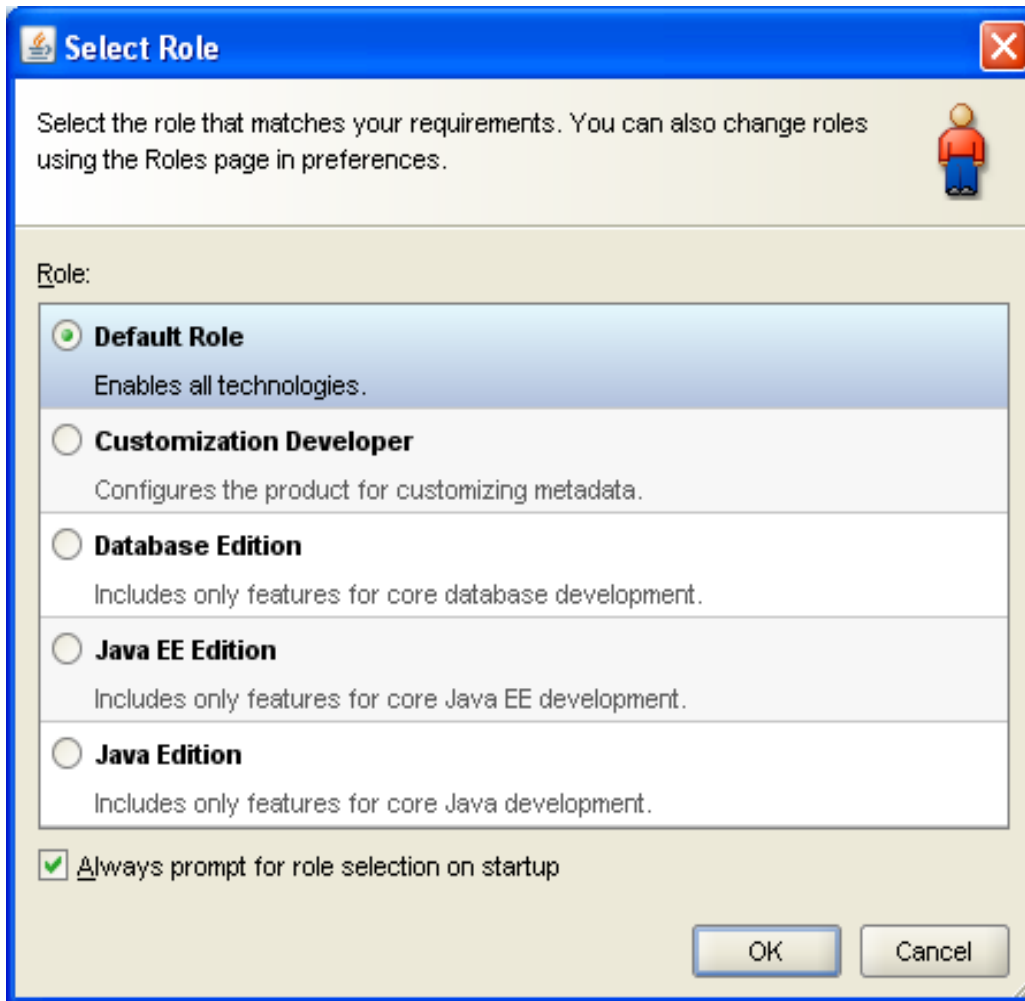
How To Create Employee With Designation Report In ADF Attractively

I am going to explain how we can create the Employee with designation report in ADF attractively.

1.Start JDeveloper by selecting **Start > Programs > Oracle Fusion Middleware 11.1.6.0.0 >JDeveloper Studio 11.1.1.6.0.**



2. In the Select Role dialog, choose “**Default Role**” and click “**OK**”.

The image shows a 'Select Role' dialog box with a blue title bar and a close button. The main area has a light beige background. At the top, there is a text instruction and a small person icon. Below this, a list of roles is shown, each with a radio button and a description. The 'Default Role' is selected. At the bottom, there is a checkbox for 'Always prompt for role selection on startup' and two buttons: 'OK' and 'Cancel'.

Select Role

Select the role that matches your requirements. You can also change roles using the Roles page in preferences.

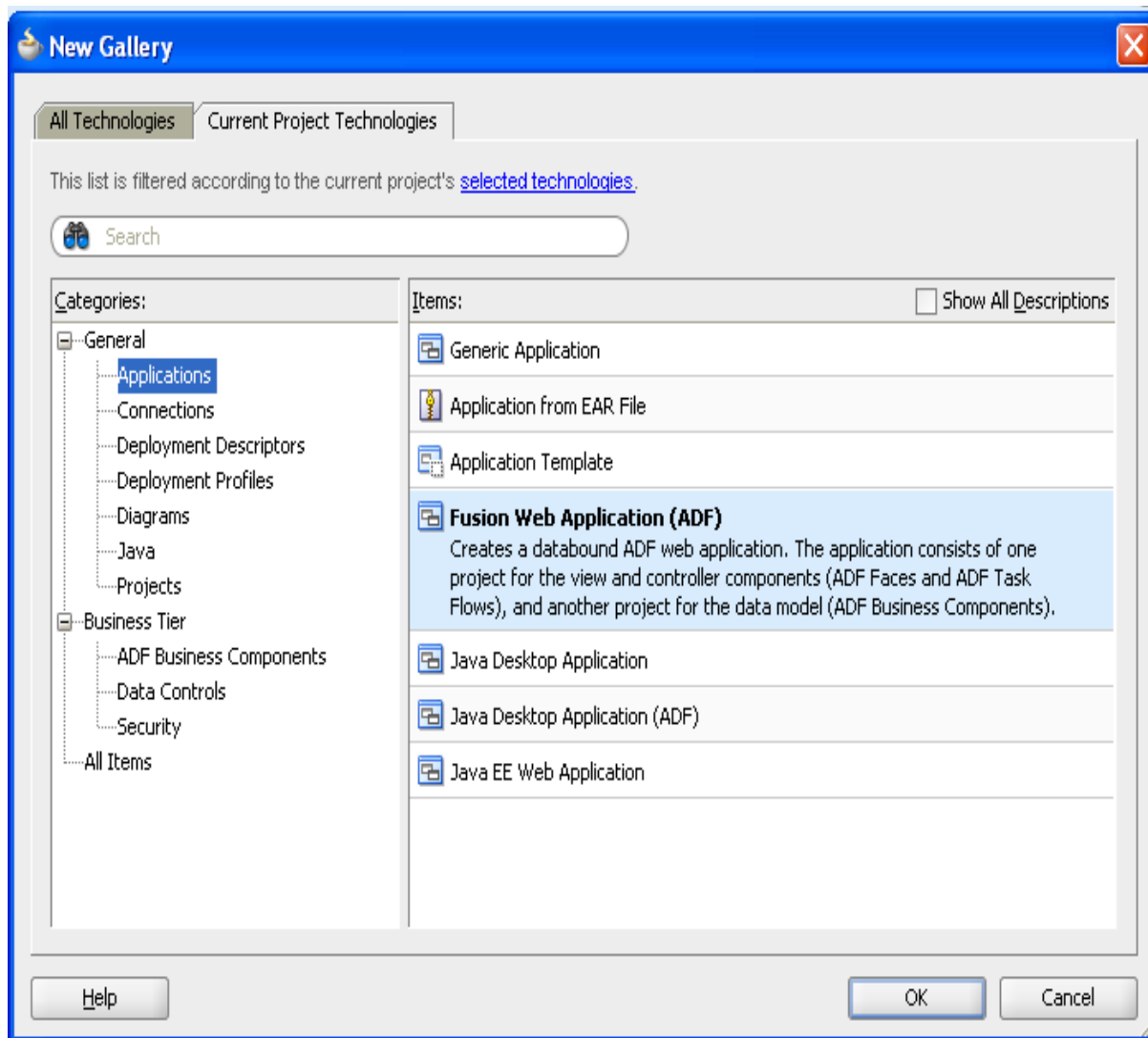
Role:

- ☒ **Default Role**
Enables all technologies.
- ☐ **Customization Developer**
Configures the product for customizing metadata.
- ☐ **Database Edition**
Includes only features for core database development.
- ☐ **Java EE Edition**
Includes only features for core Java EE development.
- ☐ **Java Edition**
Includes only features for core Java development.

☒ Always prompt for role selection on startup

OK Cancel

3. **File > New** then selecting the Applications menu item in the left side of the new dialog, select the **Fusion Web Application (ADF)** type and click “**OK**”.



4. Create the application name as EmployeeDesign click “**Next**”.

Create Fusion Web Application (ADF) - Step 1 of 5

Name your application

Application Name

Application Name: EmployeeDesign

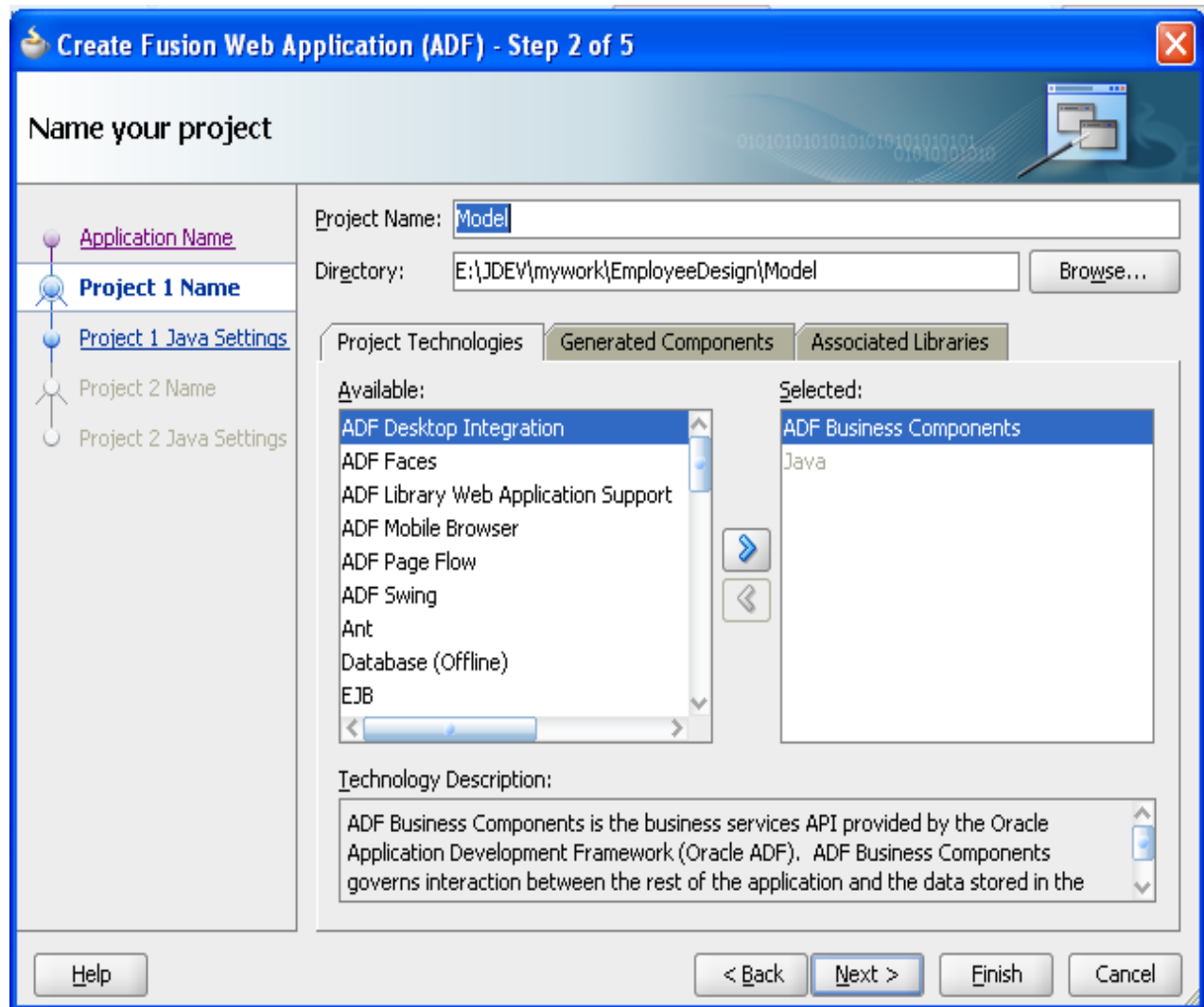
Directory:

E:\JDEV\mywork\EmployeeDesign Browse...

Application Package Prefix:

Help < Back Next > Finish Cancel

5. It will create model project click “Next”.



6. It will create javaSettings click **“Next”**.

Create Fusion Web Application (ADF) - Step 3 of 5

Configure Java settings

Your new project starts with a default package, a source root directory, and an output directory.

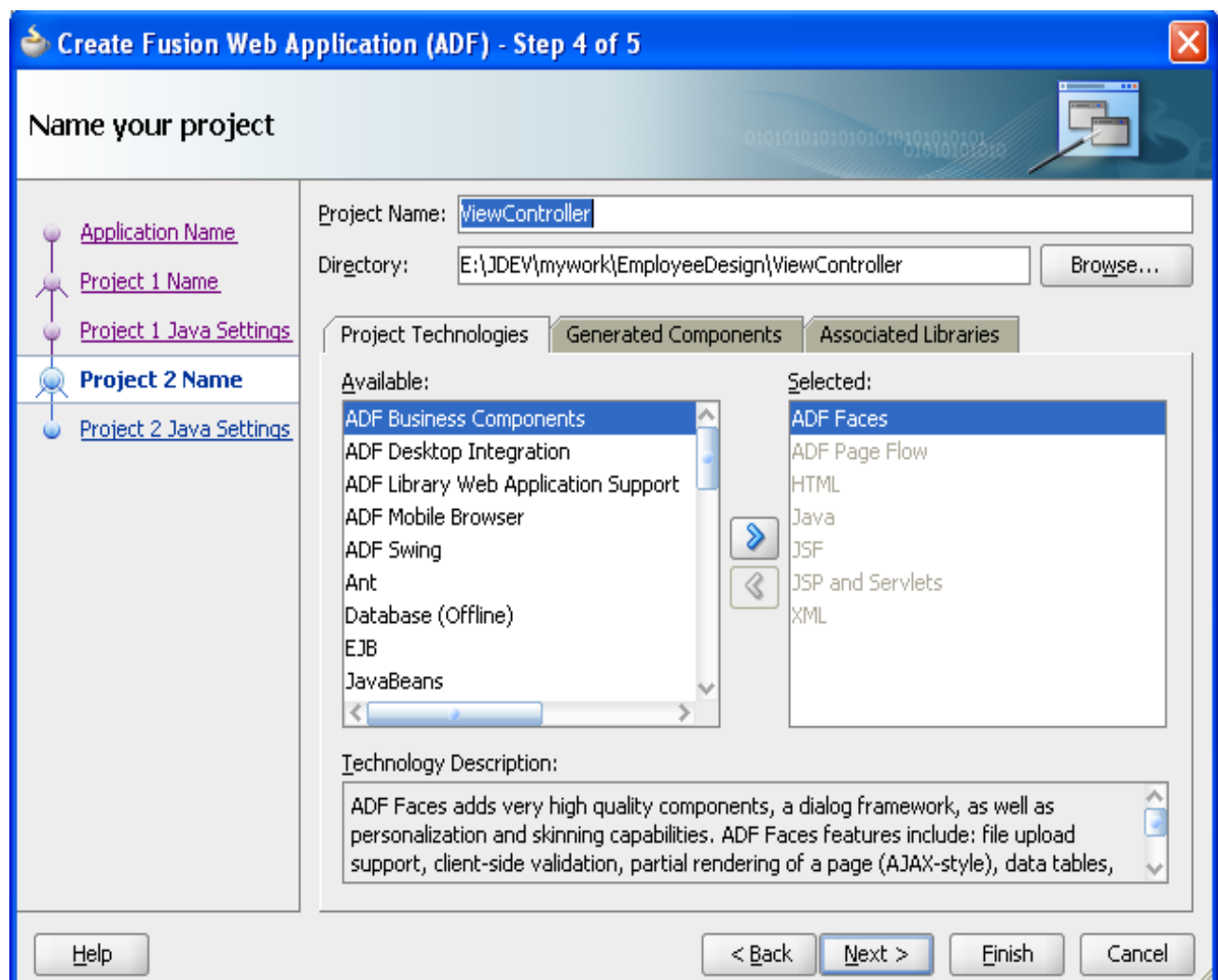
Default Package:
model

Java Source Path:
E:\JDEV\mywork\EmployeeDesign\Model\src Browse...

Output Directory:
E:\JDEV\mywork\EmployeeDesign\Model\classes Browse...

Help < Back Next > Finish Cancel

7.click “Next”.



8. Click **Finish**.

Create Fusion Web Application (ADF) - Step 5 of 5

Configure Java settings

Your new project starts with a default package, a source root directory, and an output directory.

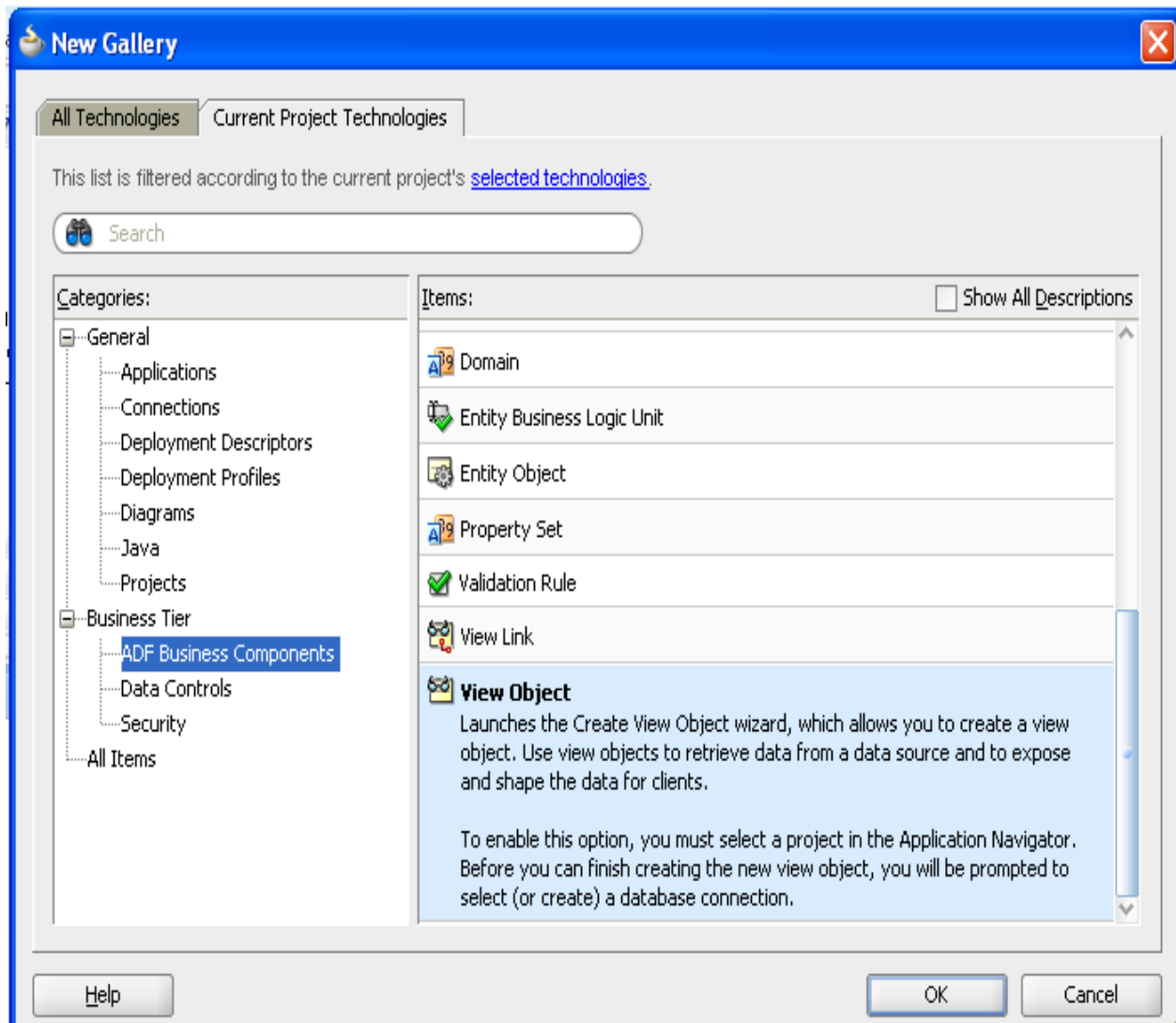
Default Package:

Java Source Path:

Output Directory:

<

9. **Right-click** on the new **Model** project and select **new**. We're going to create some business components – these are the components that interact with your database. Select **ADF Business Components** from the left side and **View Object** on the right. Then click **“OK”**.






10. In the first panel of the wizard, we're going to need to define a connection to our database. Click the **green plus symbol "+"** next to the connection drop down.

Initialize Business Components Project

This project has not yet been initialized for Business Components. After specifying the following information for your Business Components Project (.jpx file), you will be prompted to create your Business Component(s).

Specify the database connection that lets you create Business Components from existing database objects.

Connection:   

User Name:

Driver:

Connect String:

Choose the proper SQL flavor and type map that fits your application.

SQL Flavor:

Type Map:

Help **OK** **Cancel**

11. Fill in the appropriate values as shown and click “OK”. Select “Test Connection” to ensure the proper credentials are entered.

Create Database Connection

Configure a new database connection and add it to the current application (EmployeeDesign).

Create Connection In: ☒ Application Resources ☐ IDE Connections

Connection Name:

Connection Type:

Username: Role:

Password: ☒ Save Password

- Oracle (JDBC) Settings

☐ Enter Custom JDBC URL

Driver:

Host Name: JDBC Port:

☒ SID:

☐ Service Name:

Success!

12. Click “OK” again to close the connection dialog and you will be taken into the ADF BC wizard.

13. Mention the View Object Name as EmpDesig then click “Next”.

Create View Object - Step 1 of 9

Name

View objects are for joining, filtering, projecting, and sorting your business data for the specific needs of a given application task.

Package:

Name:

Display Name:

Extends:

Property Set:

Select the data source type you want to use as the basis for this view object.

☐ Updatable access through entity objects

☒ Read-only access through SQL query

☐ Rows populated programmatically, not based on a query

☐ Rows populated at design time (Static List)

14. Click Button “**Query Builder..**”

Create View Object - Step 2 of 9

Query

Enter your custom SELECT statement and click Test to check its syntax. Provide the ORDER BY clause separately.

Query Statement

Query Clauses

Order By: **Edit...**

Binding Style: **Oracle Named** **▼**

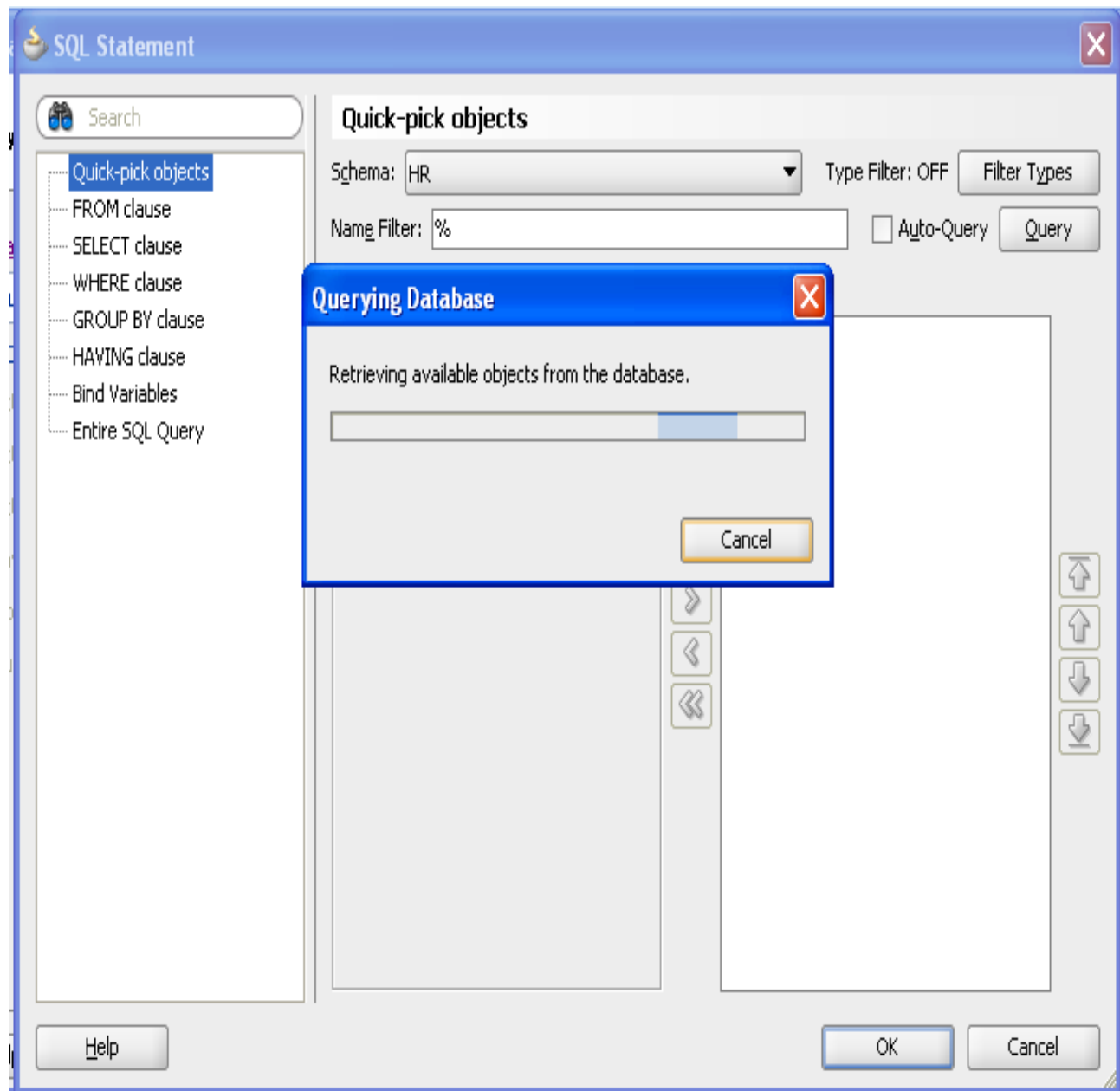
SQL Mode: **Expert** **▼**

Query Builder... **Explain Plan...** **Test**

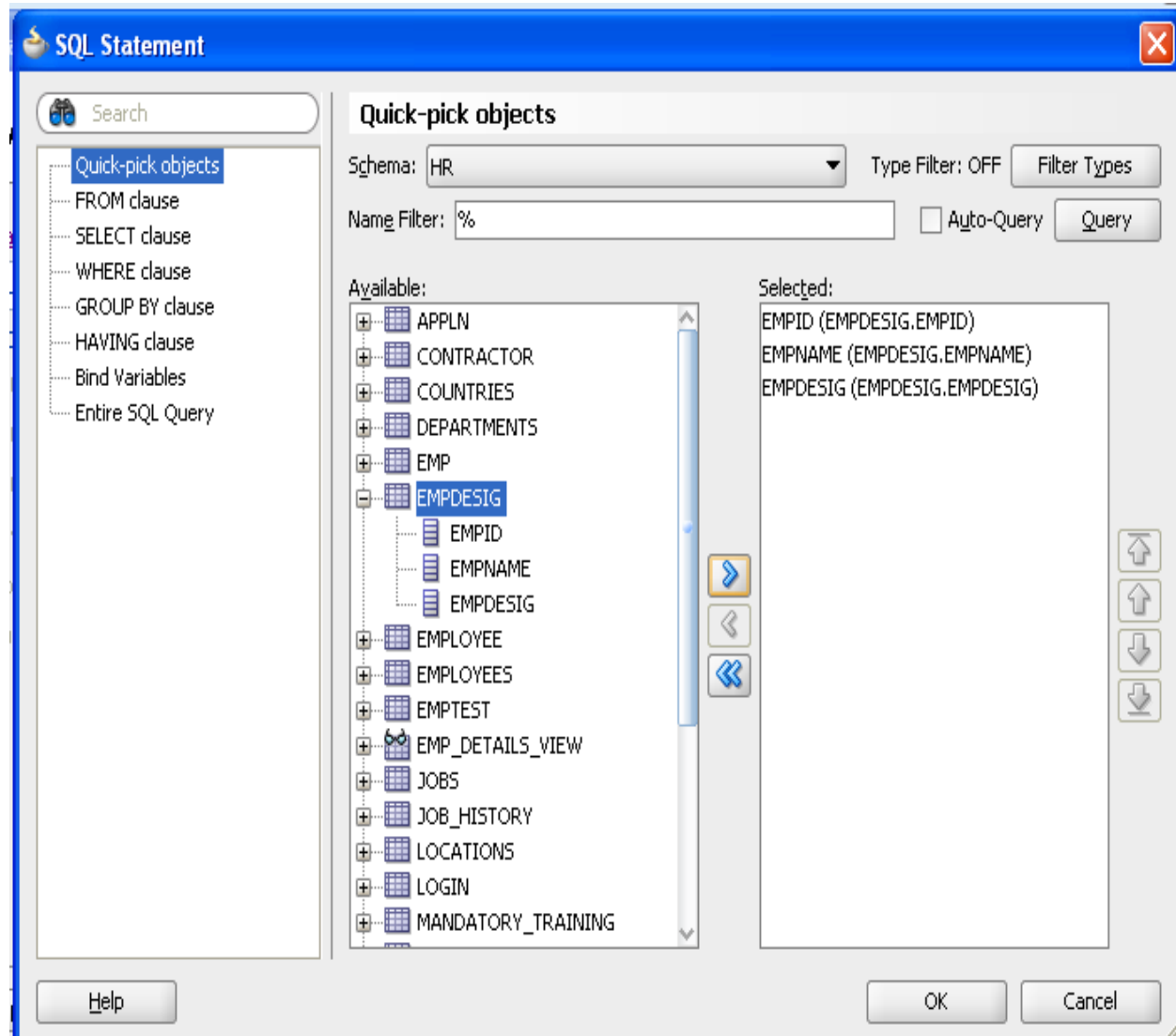
< Back **Next >** **Finish** **Cancel**

Help

15. Click **“Query”** Button.



16. Select your table which is available in oracle DB, click “**Ok**”.



17. Click “Next”.

Create View Object - Step 2 of 9

Query

Enter your custom SELECT statement and click Test to check its syntax. Provide the ORDER BY clause separately.

Query Statement

```
SELECT
    EMPDESIG.EMPID EMPID,
    EMPDESIG.EMPNAME EMPNAME,
    EMPDESIG.EMPDESIG EMPDESIG
FROM
    EMPDESIG
```

Query Clauses

Order By: **Edit...**

Binding Style: **Oracle Named**

SQL Mode: **Expert**

18. Just click “Next” button for step3 to step7.

19. In Step 8 Enable "Application Module" check box then click "Next".

Create View Object - Step 8 of 9

Application Module

Select the checkbox to add an instance of this view object to an application module. If the specified application module does not exist, it will be created.

☒ **Application Module**

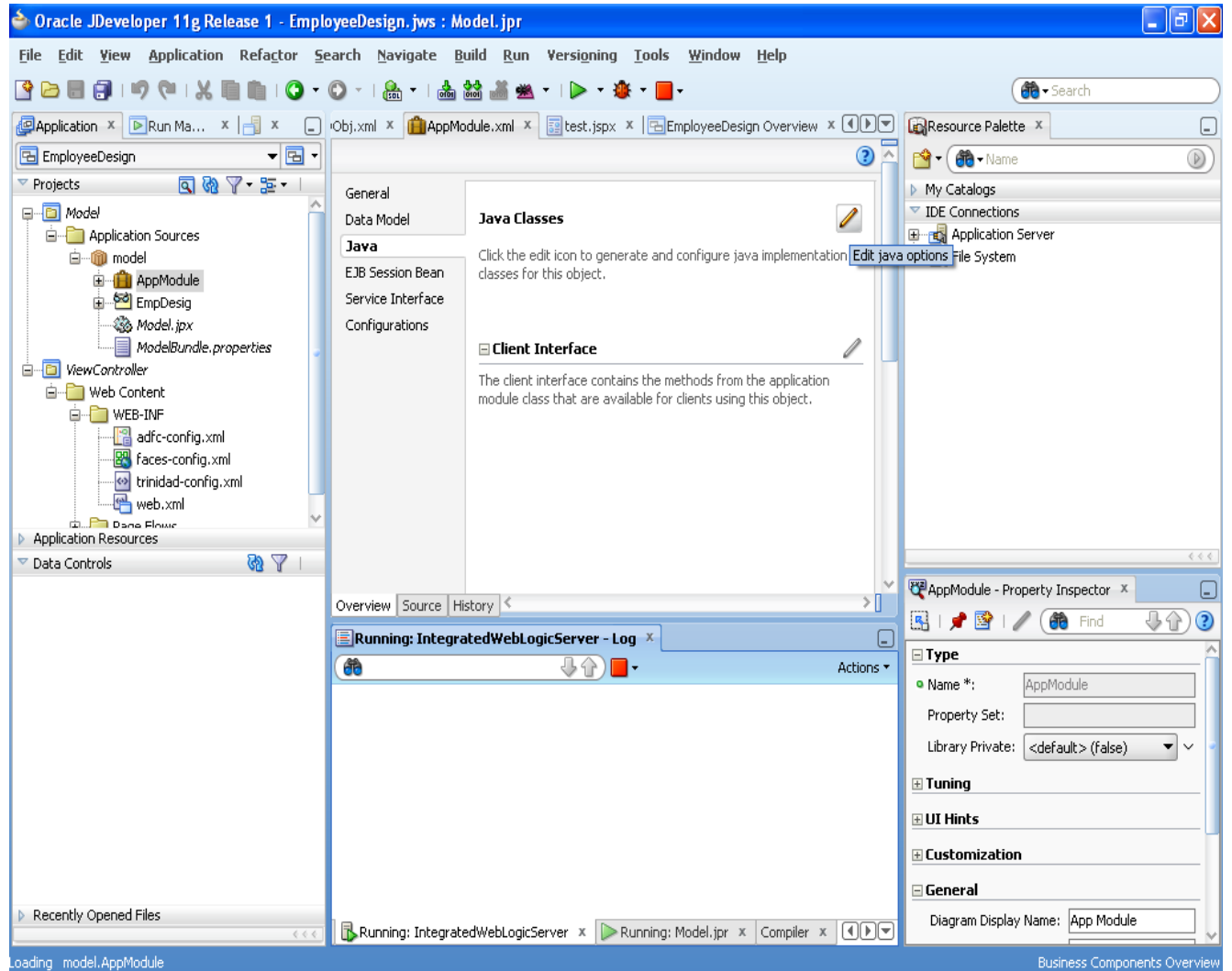
Package:

Name:

Java

- Application Module**
- Summary

20. We will now edit the **AppModule**, Double click this item to open it for editing. Select “Java” Tab then click “Edit” (Pencil Symbol).



21.Enable the check Box Generate “**Application Module Class**” ,then click “OK”.



To add the following java method in your AppModuleImpl.java file

```
public List<String> prepareList(){  
    List<String> list=new ArrayList<String>();  
    ViewObjectImpl impl=getEmpDesig1();  
    impl.executeQuery();  
    while(impl.hasNext()){  
        Row row=impl.next();  
        String desig=row.getAttribute("Empdesig").toString();  
        if(desig.equalsIgnoreCase("Manager")){  
            desig+="red";  
        }  
        if(desig.equalsIgnoreCase("Employee")){  
            desig+="blue";  
        }  
        if(desig.equalsIgnoreCase("Trainee")){  
            desig+="green";  
        }  
        if(desig.equalsIgnoreCase("Associate")){  
            desig+="orange";  
        }  
        if(desig.equalsIgnoreCase("Developer")){  
            desig+="yellow";  
        }  
        list.add(row.getAttribute("Empname").toString()+" "+desig);  
    }  
}
```

```

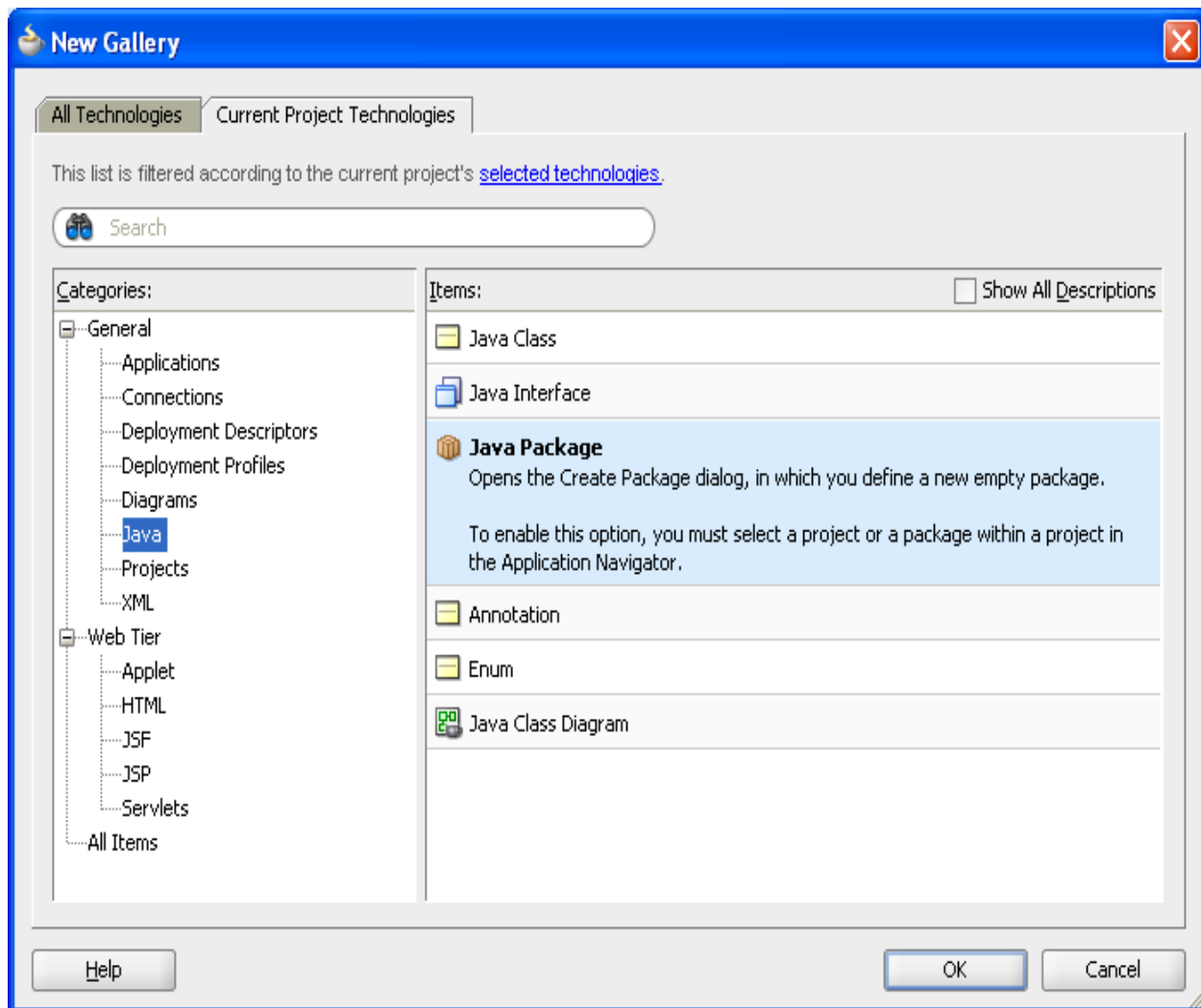
    }

    return list;
}

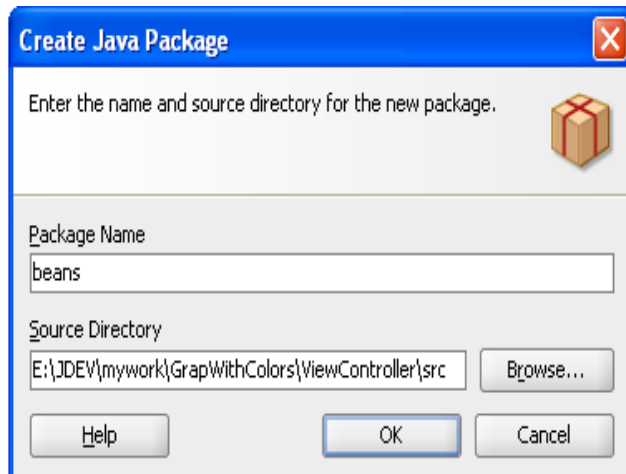
```

Existing code explains each designation having different colors.

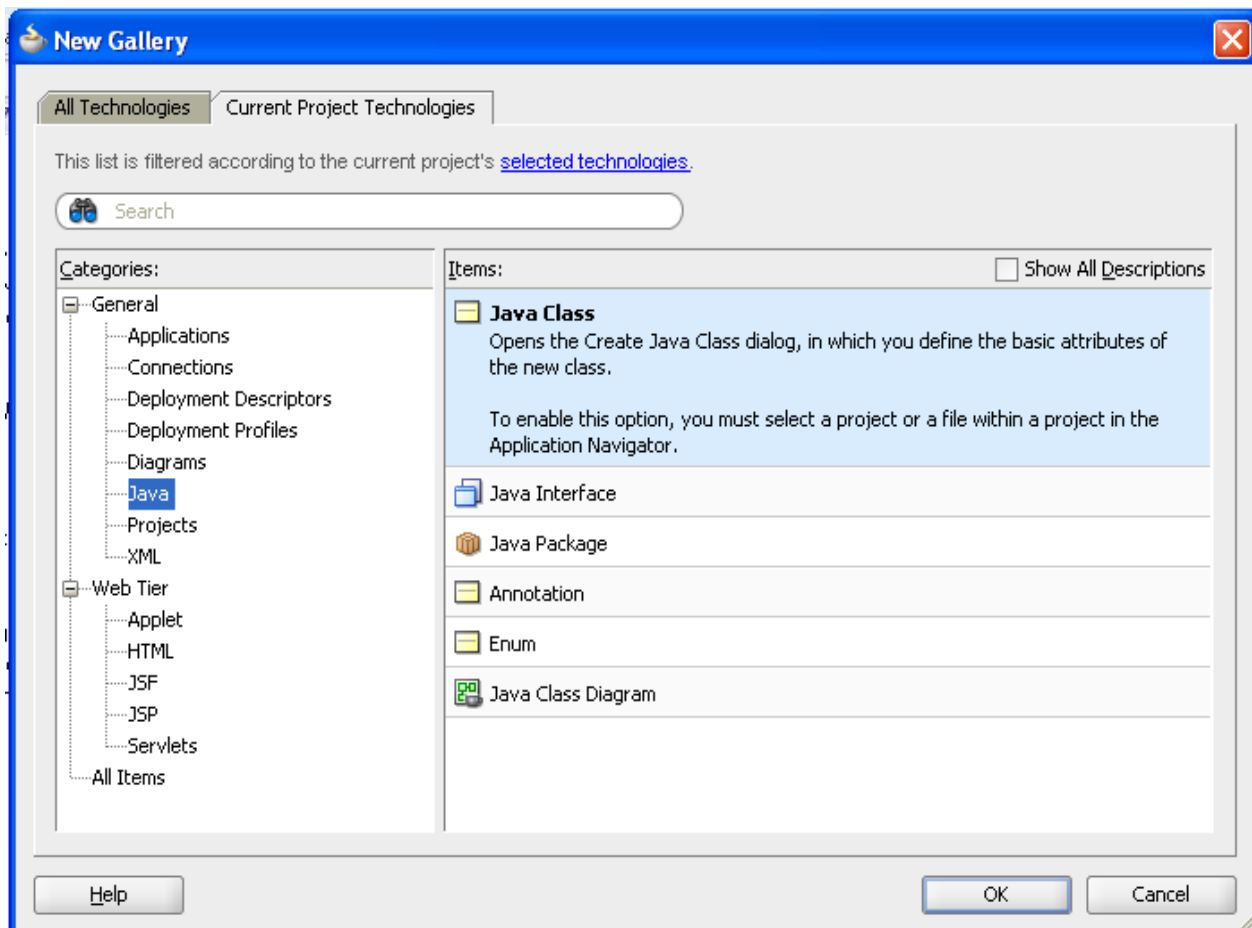
22.Right-click on the new **View Controller** project and select **new**. We're going to create a Bean Class to interact UI. Select **java** from the left side and **Java Package** on the right. Then click "**OK**".



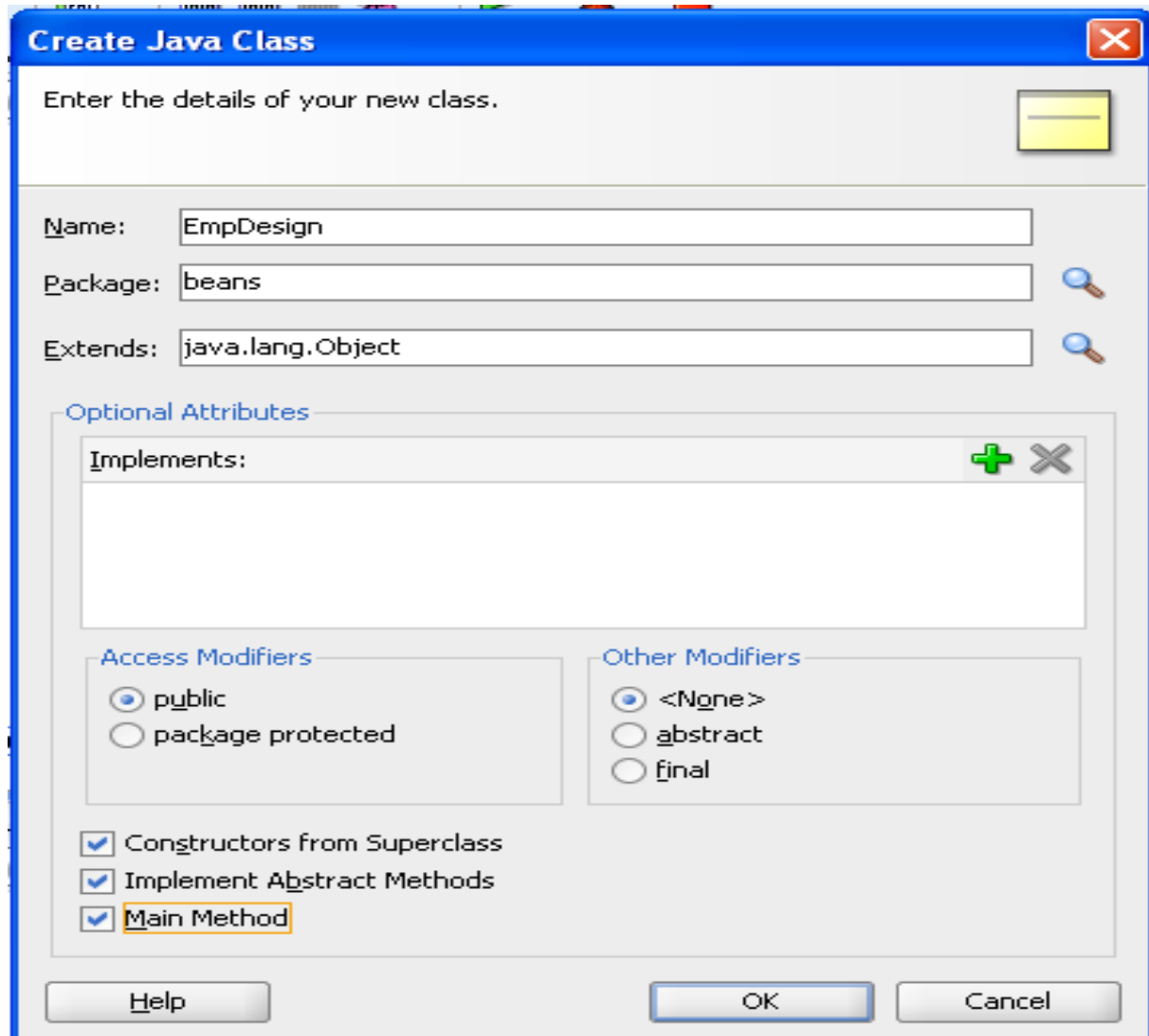
23. Mention The package name as beans then click “OK”.



24. **Right-click** on the new **beans** package and select **new..** Select **java** from the left side and **Java Class** on the right. Then click “OK”.



25. Mention your class name as GraphBean then click "OK".



The image shows a 'Create Java Class' dialog box with a blue title bar and a close button in the top right corner. The main area is light gray and contains the text 'Enter the details of your new class.' followed by a small yellow icon. Below this, there are three text input fields: 'Name:' with 'EmpDesign', 'Package:' with 'beans', and 'Extends:' with 'java.lang.Object'. To the right of the 'Package' and 'Extends' fields are magnifying glass icons. Below these fields is a section titled 'Optional Attributes' with a blue header. Inside this section is a large text area labeled 'Implements:' with a green plus icon and a gray X icon in the top right corner. Below the 'Implements' text area are two sub-sections: 'Access Modifiers' and 'Other Modifiers'. The 'Access Modifiers' section has two radio buttons: 'public' (selected) and 'package protected'. The 'Other Modifiers' section has three radio buttons: '<None>' (selected), 'abstract', and 'final'. Below these sections are three checked checkboxes: 'Constructors from Superclass', 'Implement Abstract Methods', and 'Main Method'. At the bottom of the dialog are three buttons: 'Help', 'OK', and 'Cancel'.

Create Java Class

Enter the details of your new class.

Name: EmpDesign

Package: beans

Extends: java.lang.Object

Optional Attributes

Implements:

Access Modifiers

☒ public

☐ package protected

Other Modifiers

☒ <None>

☐ abstract

☐ final

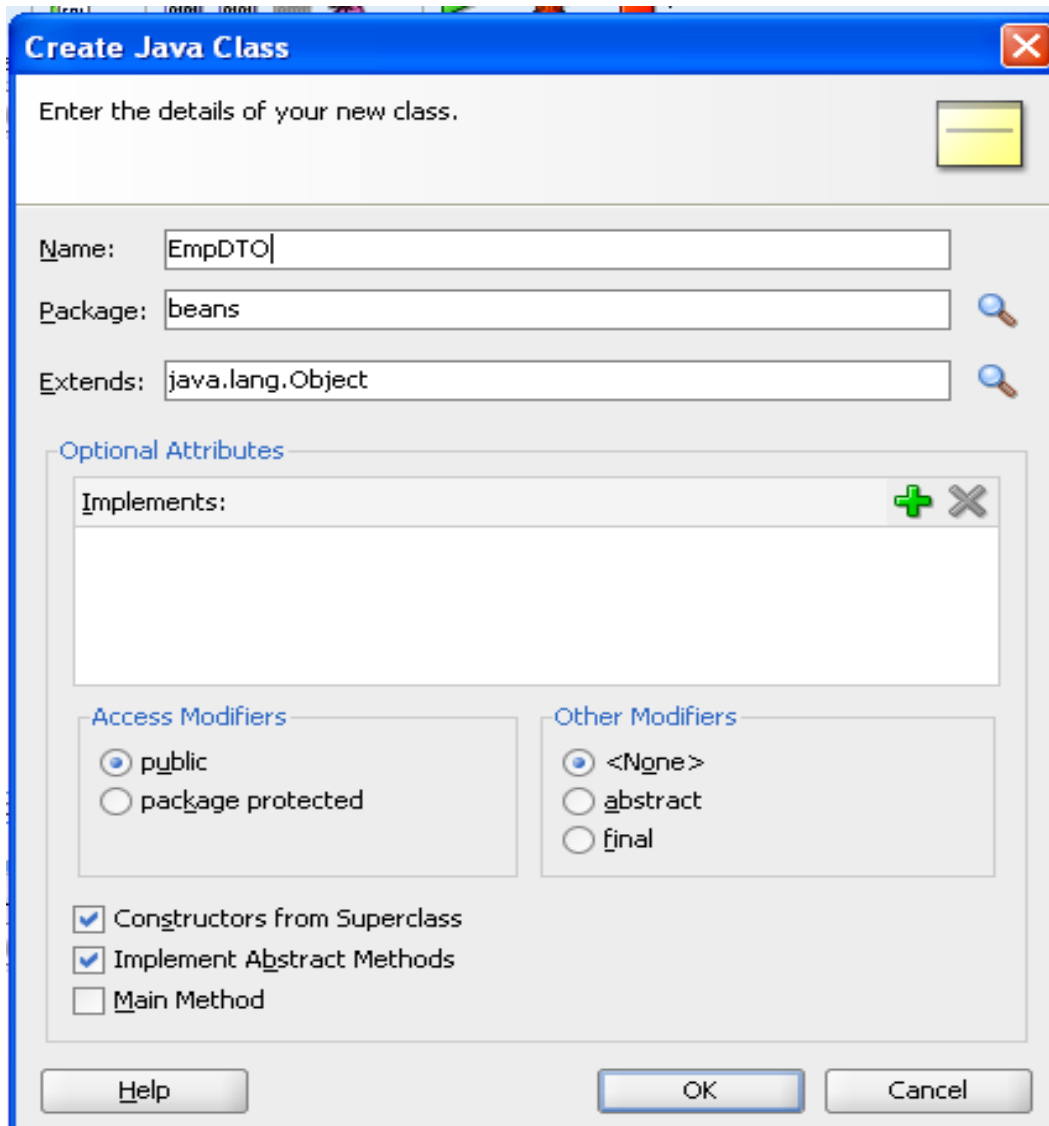
☒ Constructors from Superclass

☒ Implement Abstract Methods

☒ Main Method

Help OK Cancel

26. In same package we need to create DTO class name as EmpDTO.



The image shows a 'Create Java Class' dialog box with a blue title bar and a close button (X) in the top right corner. The main area is light gray and contains the text 'Enter the details of your new class.' followed by a yellow icon. Below this are three text input fields: 'Name:' with 'EmpDTO', 'Package:' with 'beans', and 'Extends:' with 'java.lang.Object'. To the right of the 'Package' and 'Extends' fields are magnifying glass icons. Below these fields is a section titled 'Optional Attributes' in blue. It contains a large text area labeled 'Implements:' with a green plus icon and a gray X icon to its right. Below the 'Implements' area are two groups of radio buttons. The first group, 'Access Modifiers', has 'public' selected and 'package protected' as an option. The second group, 'Other Modifiers', has '<None>' selected, with 'abstract' and 'final' as other options. At the bottom of the 'Optional Attributes' section are three checkboxes: 'Constructors from Superclass' (checked), 'Implement Abstract Methods' (checked), and 'Main Method' (unchecked). At the very bottom of the dialog are three buttons: 'Help', 'OK', and 'Cancel'.

Create Java Class

Enter the details of your new class.

Name: EmpDTO

Package: beans

Extends: java.lang.Object

Optional Attributes

Implements:

Access Modifiers

- ☒ public
- ☐ package protected

Other Modifiers

- ☒ <None>
- ☐ abstract
- ☐ final

☒ Constructors from Superclass

☒ Implement Abstract Methods

☐ Main Method

Help OK Cancel

27.To add following code in EmpDTO.java File.

```
package beans;

public class EmpDTO {

    public EmpDTO() {

        super();

    }

    private String emp;

    private String des;

    private String color;

    public void setEmp(String emp) {

        this.emp = emp;

    }

    public String getEmp() {

        return emp;

    }

    public void setDes(String des) {

        this.des = des;

    }

    public String getDes() {

        return des;

    }

    public void setColor(String color) {

        this.color = color;
```

```

    }

    public String getColor() {

        return color;

    }

}

```

28.To add following code in EmpDesign.java

```
package beans;
```

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
import model.AppModuleImpl;
```

```
import oracle.jbo.client.Configuration;
```

```
public class EmpDesign {
```

```
    public EmpDesign() {
```

```
        super();
```

```
    }
```

```
    public static void main(String[] args) {
```

```
        EmpDesign empDesign = new EmpDesign();
```

```
        empDesign.getList();
```

```
    }
```

```
    private List<EmpDTO> empList1 = new ArrayList<EmpDTO>();
```

```
    public String getList() {
```

```

AppModuleImpl imp =
    (AppModuleImpl)Configuration.createRootApplicationModule("model.AppModule",
        "AppModuleLocal");

List<String> list = imp.prepareList();

Configuration.releaseRootApplicationModule(imp, true);

System.out.println(list);

List<EmpDTO> list1 = new ArrayList<EmpDTO>();

for (int i = 0; i < list.size(); i++) {

    String[] st = list.get(i).split(",");

    EmpDTO dto = new EmpDTO();

    dto.setEmp(st[0]);

    dto.setDes(st[1]);

    dto.setColor(st[2]);

    list1.add(dto);

}

this.setEmpList1(list1);

return "True";

}

public void setEmpList1(List<EmpDTO> empList1) {

    this.empList1 = empList1;

}

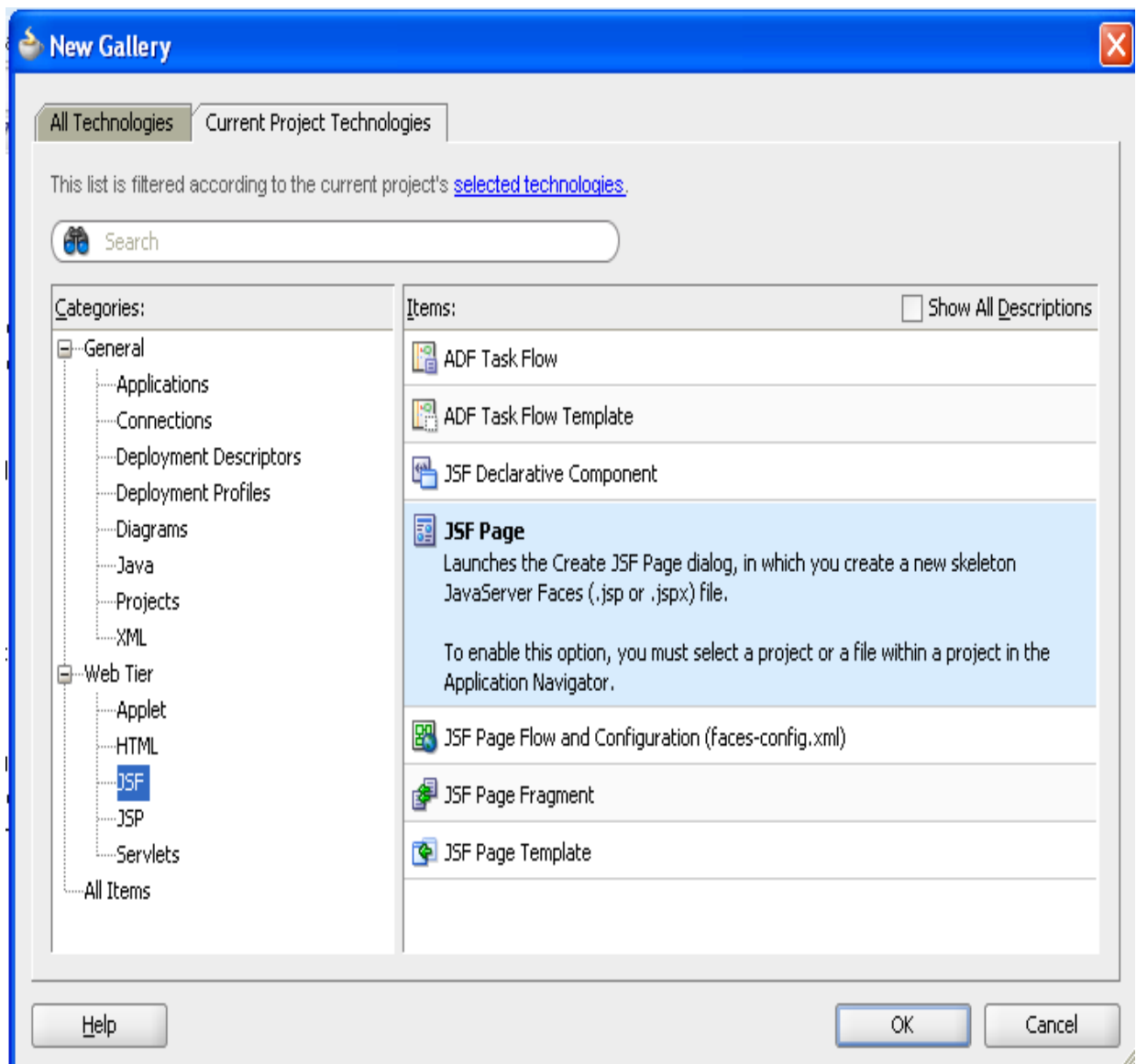
```

```
public List<EmpDTO> getEmpList1() {  
    return empList1;  
}  
}
```

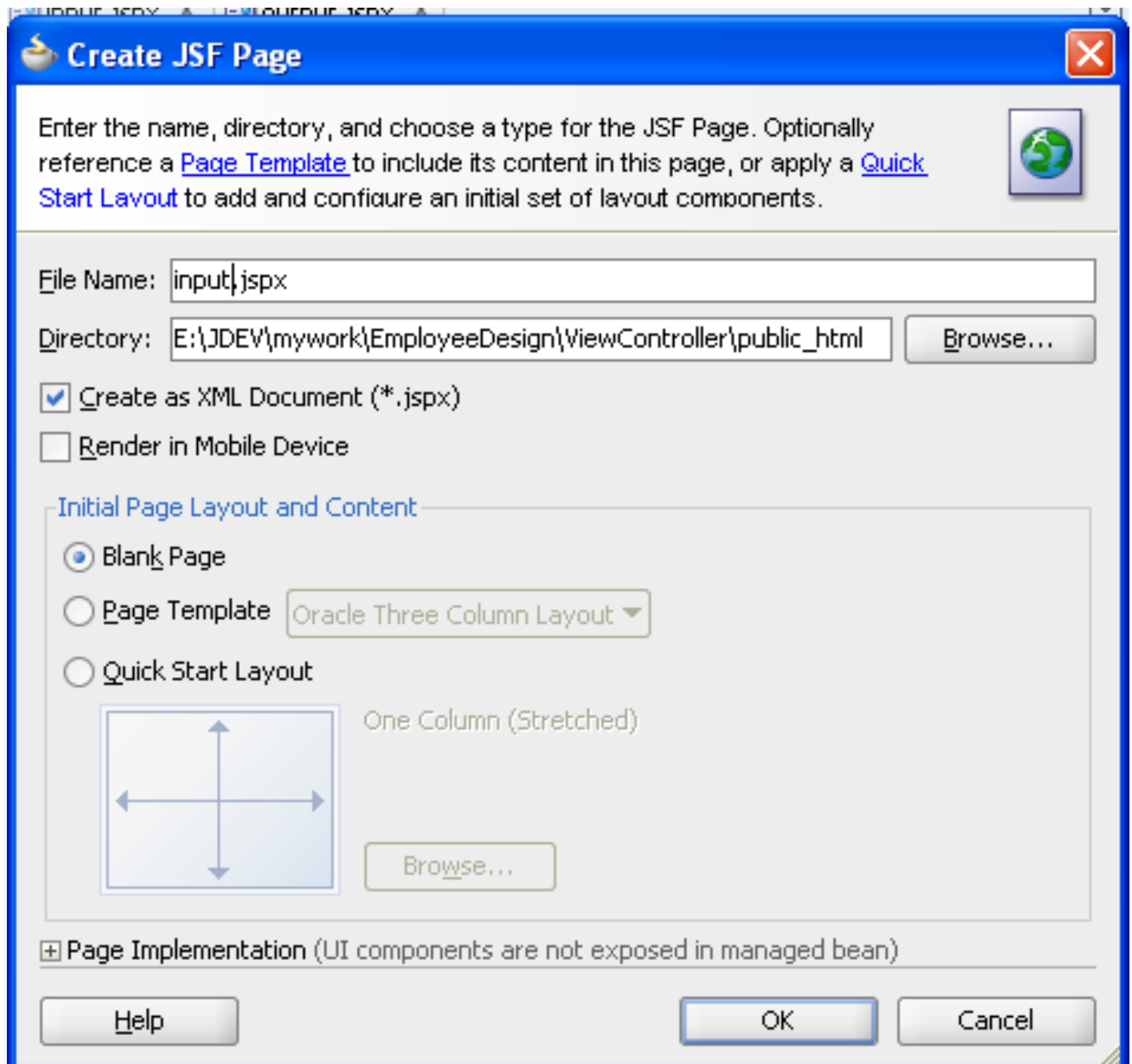
29.To add following code in adfconfig.xml

```
<?xml version="1.0" encoding="windows-1252" ?>  
  
<adfc-config xmlns="http://xmlns.oracle.com/adf/controller" version="1.2">  
  
    <managed-bean id="__1">  
  
        <managed-bean-name id="__4">emp</managed-bean-name>  
  
        <managed-bean-class id="__2">beans.EmpDesign</managed-bean-class>  
  
        <managed-bean-scope id="__3">request</managed-bean-scope>  
  
    </managed-bean>  
  
</adfc-config>
```

30. **Right-click** on the new **Web Content** Folder and select **new**. Select **jsff** from the left side and **JSF Page** on the right. Then click **“OK”**.



31. Mention The Page name as input.jsx then click "OK".



The image shows a 'Create JSF Page' dialog box with a blue title bar and a close button (X) in the top right corner. The main area has a light gray background. At the top, there is a text instruction: 'Enter the name, directory, and choose a type for the JSF Page. Optionally reference a [Page Template](#) to include its content in this page, or apply a [Quick Start Layout](#) to add and configure an initial set of layout components.' To the right of this text is a small icon of a globe with a green arrow. Below the instruction, there are three input fields: 'File Name:' with the text 'input.jsx', 'Directory:' with the text 'E:\JDEV\mywork\EmployeeDesign\ViewController\public_html', and a 'Browse...' button to the right. Below these fields are two checkboxes: 'Create as XML Document (*.jspx)' which is checked, and 'Render in Mobile Device' which is unchecked. A section titled 'Initial Page Layout and Content' contains three radio buttons: 'Blank Page' (selected), 'Page Template' (with a dropdown menu showing 'Oracle Three Column Layout'), and 'Quick Start Layout'. Below the radio buttons is a diagram of a page layout with a central rectangle and four arrows pointing outwards (up, down, left, right). To the right of the diagram is the text 'One Column (Stretched)' and a 'Browse...' button. At the bottom of the dialog, there is a checkbox labeled '+ Page Implementation (UI components are not exposed in managed bean)'. The bottom of the dialog features three buttons: 'Help', 'OK', and 'Cancel'.

Create JSF Page

Enter the name, directory, and choose a type for the JSF Page. Optionally reference a [Page Template](#) to include its content in this page, or apply a [Quick Start Layout](#) to add and configure an initial set of layout components.

File Name:

Directory:

☒ Create as XML Document (*.jspx)


☐ Render in Mobile Device

Initial Page Layout and Content

☒ Blank Page

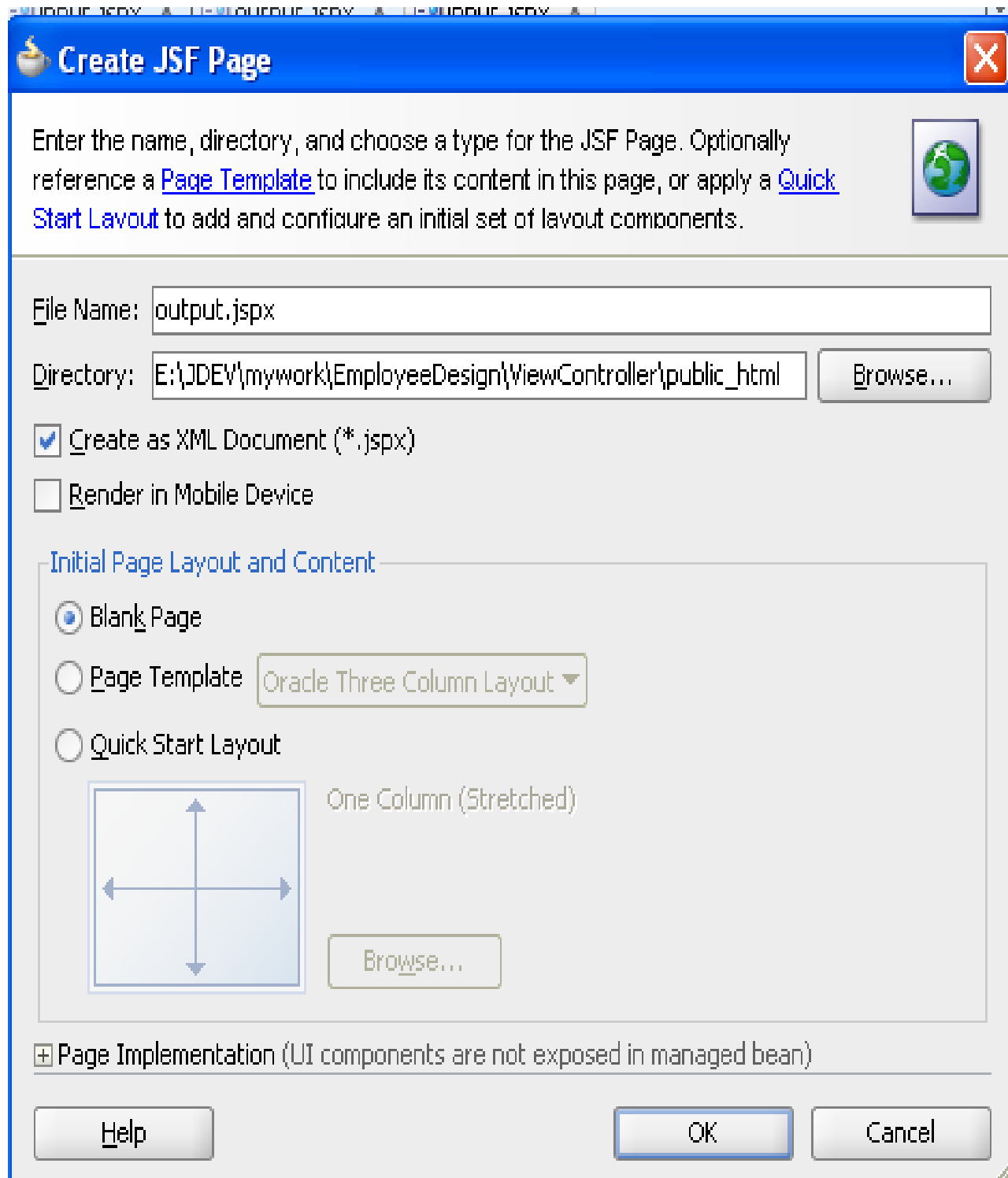
☐ Page Template



☐ Quick Start Layout


 One Column (Stretched)

☒ Page Implementation (UI components are not exposed in managed bean)

32 Create another jsf page, Metion The Page name as output.jspx then click “OK”.



 **Create JSF Page** 

Enter the name, directory, and choose a type for the JSF Page. Optionally reference a [Page Template](#) to include its content in this page, or apply a [Quick Start Layout](#) to add and configure an initial set of layout components. 

File Name:

Directory:

☒ Create as XML Document (*.jspx)

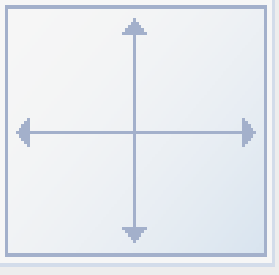
☐ Render in Mobile Device

Initial Page Layout and Content

☒ Blank Page

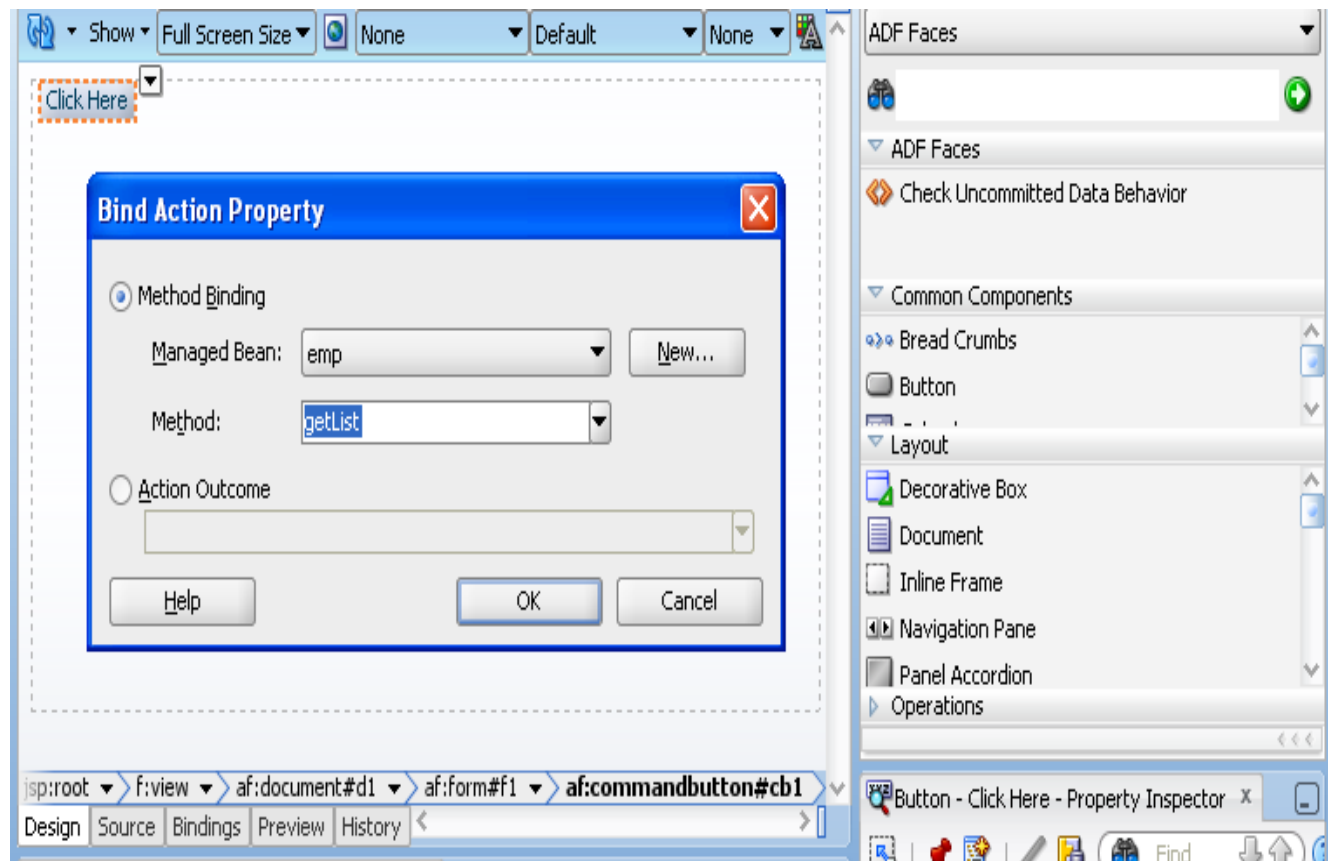
☐ Page Template

☐ Quick Start Layout

 One Column (Stretched)

☒ Page Implementation (UI components are not exposed in managed bean)

33. Drag and drop the command button in input.jspx, next double click the button select alias name of the bean and method.



(Or)

To add following code in input.jspx page

```
<af:commandButton text="Click Here" id="cb1" action="#{emp.getList}"/>
```

34. To add following code in output.jspx page:

```
<?xml version='1.0' encoding='UTF-8'?>
```

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
```



```

xmlns:f="http://java.sun.com/jsf/core"

xmlns:h="http://java.sun.com/jsf/html"

xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
<jsp:directive.page contentType="text/html; charset=UTF-8"/>
<f:view>

<af:document id="d1">

<af:form id="f1">

<af:forEach items="#{emp.empList1}" var="dto">

<af:decorativeBox id="db21"

    inlineStyle="width:180px; border-color:Gray; height:50px;">

<f:facet name="center">

<af:panelGroupLayout id="pgl41" layout="vertical">

<af:panelGroupLayout id="pgl71" layout="horizontal"

    halign="start" >

<af:outputText value="#{dto.emp}"

    id="ot18"

    inlineStyle="color:rgb(0,99,148); font-family:Arial, Helvetica, sans-serif; font-size:small;
text-align:left; vertical-align:middle; font-weight:bold; border-color:Gray; "/>

</af:panelGroupLayout>

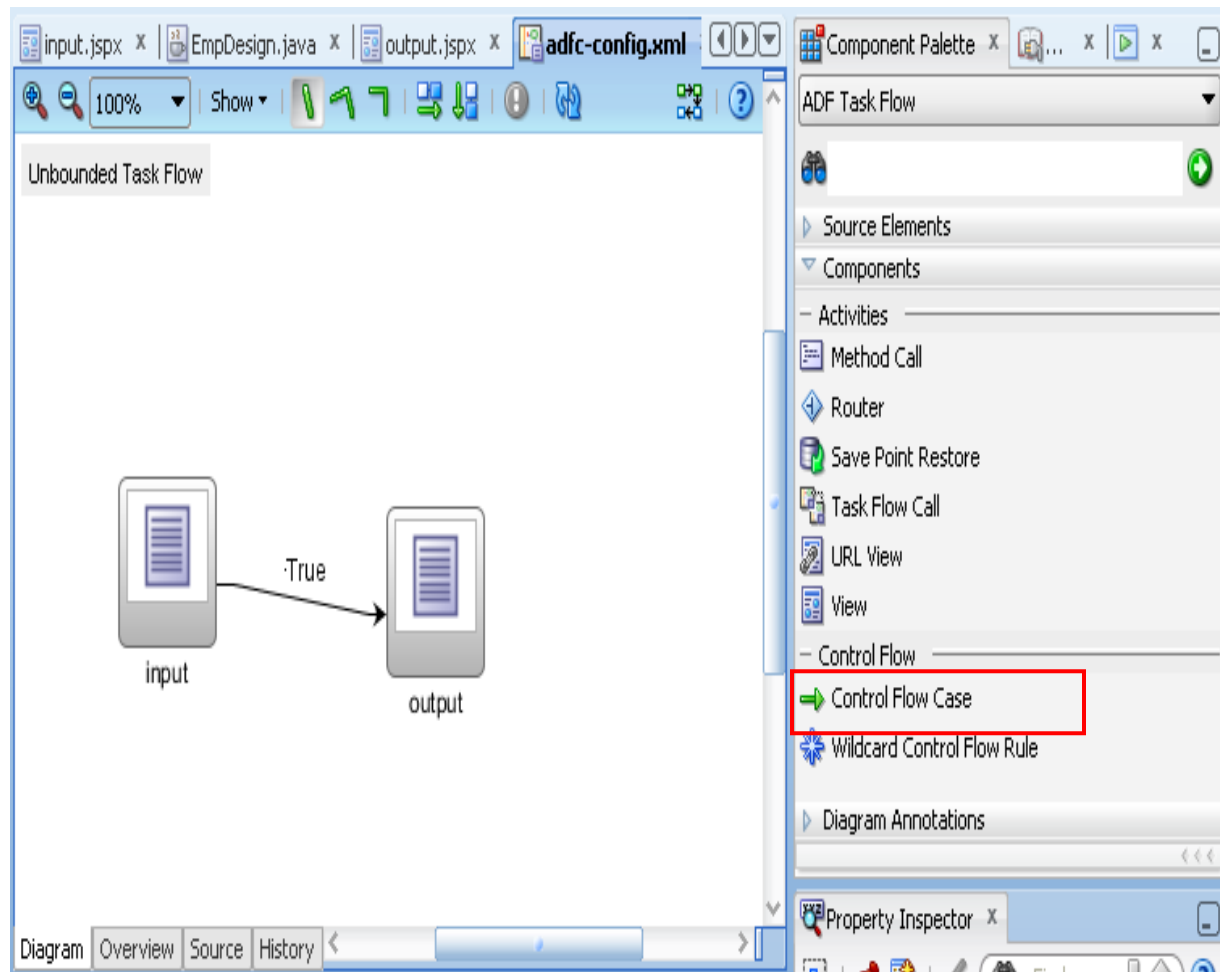
<af:spacer width="1000000" height="5" id="s25"/>

<af:panelGroupLayout id="pgl721" layout="horizontal"    halign="end" >

```

```
<af:outputText value="#{dto.des}" id="ot25"
                inlineStyle=" font-family:Arial, Helvetica, sans-serif; font-size:xx-small; font-
weight:bold; background-color:#{dto.color};"/>
    </af:panelGroupLayout>
</af:panelGroupLayout>
</f:facet>
</af:decorativeBox>
</af:forEach>
</af:form>
</af:document>
</f:view>
</jsp:root>
```

35. In adfconfig.xml to drag and drop the input.jspx and output.jspx as shown below. then in component palette drag the Control Flow Case to connect the two pages and set the value for From Out Come as "True".



OutPut:

1.Now run your input.jspx page



2.Click "Click Here" Button.



**Thank you,Keep Breathing &
Keep Learning**