

Using Lightning Web Components in Salesforce Experiences

Instructions



TRAILHEAD ACADEMY

Set Up the Bear Project

Tasks:

1. Set Up Your Trailhead Playground.
2. Set Up the Bear Project.

-
1. Open a command prompt.
 2. Clone the bear LWC App repository:

```
git clone https://github.com/sfdc-cdev/Bear.git
```

3. Navigate to the new Bear directory:

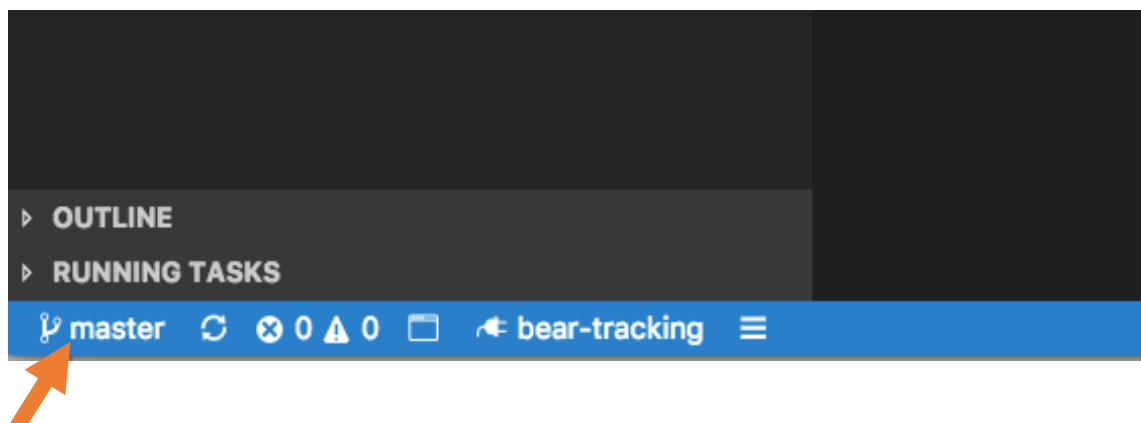
```
cd Bear
```

4. Open VS Code:

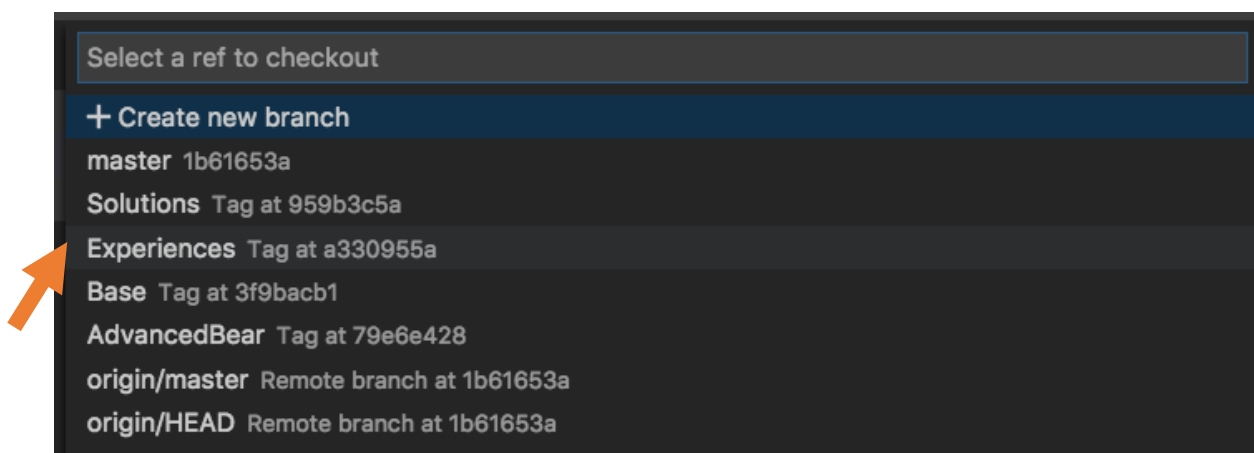
```
code .
```

5. Checkout the branch named **Experiences**:

- Click on **master**:



- Select **Experiences**:



6. Authorize your Trailhead Playground with the Salesforce CLI, save it with a df19th_experiences alias and set the current user as the default user:

```
sfdx force:auth:web:login -s -a experiences
```

7. When a browser window with the Salesforce login page opens, enter your Trailhead Playground credentials.

Note: If you haven't got a Playground or if you don't know your username and password, follow the instructions in section **Set up Your Trailhead Playground** here: https://sfdc.co/LWC_HelloWorld

8. Deploy the app code to the org.

```
sfdx force:source:deploy -p force-app/main/default
```

9. Assign the Ursus Park User permission set to the current user.

```
sfdx force:user:permset:assign -n Ursus_Park_User
```

10. Import the sample data.

```
sfdx force:data:tree:import -p data/plan.json
```

11. Open the org in a browser.

```
sfdx force:org:open
```

Exercise 1: Building Lightning Pages with Components and App Builder

Tasks:

1. Create the Bears Worldwide Component.
 2. Define a Mobile Lightning Page with App Builder.
 3. Add your Custom Component to the Lightning Page.
 4. Add a Chatter Feed.
 5. Add a Filter List.
 6. Activate Your App.
-

1. Define a Bears Worldwide component.
 - A. From VS Code, open **force-app | main | default** and right-click on **lwc**. Select the following option:

```
>SFDX: Create Lightning Web Component
```

- B. Enter the following information:

- **Filename:** bearsWorldwide

- C. In **bearsWorldwide.js-meta.xml**, set **isExposed** to true, add the **Lightning__AppPage** target, and allow the **listView** and **markersTitle** attributes to be controlled at design time:

```
<isExposed>true</isExposed>
<targets>
  <target>lightning__AppPage</target>
</targets>
<targetConfigs>
  <targetConfig targets="lightning__AppPage">
    <property name="listView" label="List View"
      type="String" datasource="visible,hidden" />
    <property name="markersTitle" label="List Title (if
      visible)" type="String" />
  </targetConfig>
</targetConfigs>
```

- D. Replace the content of **bearsWorldwide.html** with the following markup:

```

<template>
  <lightning-card>
    <lightning-map>
      map-markers={mapMarkers}
      markers-title={markersTitle}
      list-view={listView}
      zoom-level="2"></lightning-map>
    </lightning-card>
  </template>

```

- E. Copy the following and paste it into the **bearsWorldwide.js** resource of bearsWorldwide component:

```

import { LightningElement, api, track, wire } from 'lwc';
import getAllBears from
 '@salesforce/apex/BearController.getAllBears';

export default class BearTab extends LightningElement {
  @track mapMarkers;
  @api markersTitle = 'Bears Worldwide';
  @api listView='visible';

  @wire(getAllBears)
  wired_getLocations({ error, data }) {
    this.mapMarkers = [];
    if (data) {
      data.forEach(loc => {
        this.mapMarkers.push({

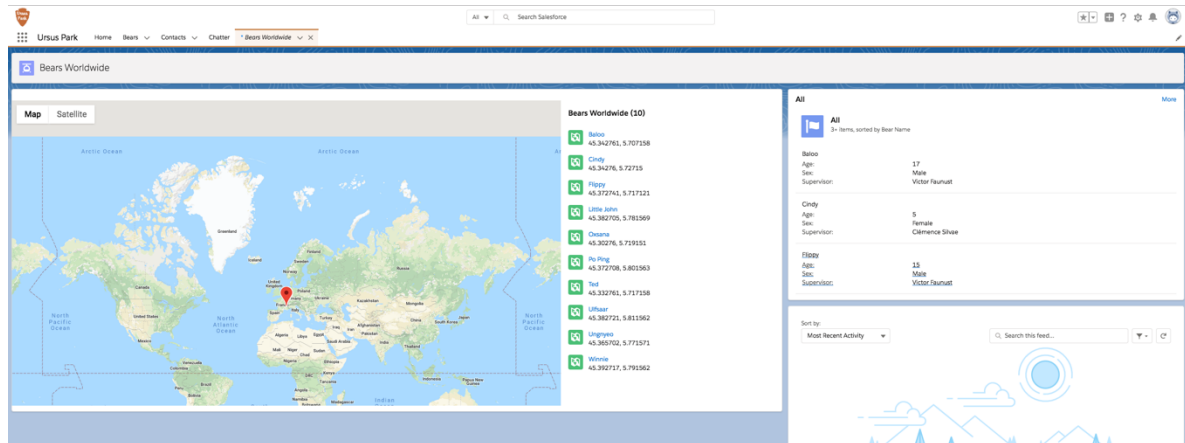
          location: {
            Latitude: loc.Location__Latitude__s,
            Longitude: loc.Location__Longitude__s
          },

          description: 'Bears around the world.',
          title: loc.Name,
        });
      });
    } else if (error) {
      this.error = error;
    }
  }
}

```

2. Define a Mobile Lightning Page with App Builder.
 - A. Deploy the updated code to the org. Right-click the default folder and click **SFDX: Deploy Source to Org**.
 - B. Return to your org and select **Setup | User Interface | Lightning App Builder**.
 - C. Click the **New** button.
 - D. Select **App Page** and Click **Next**.
 - E. Enter a Label of **Bears Worldwide** and click **Next**.
 - F. Select the **Main Region and Right Sidebar** layout and click **Finish**.

3. Add your Custom Component to the Lightning Page.
 - A. You should see your **bearsWorldwide** component appear under the Custom heading. Click, drag, and drop the bearsWorldwide onto the design canvas into the left box.
 - B. Experiment with toggling the values of the List View and List Title attributes in the bearsWorldwide component's design form. What happens if you change the value of List View to 'hidden'? Do you have to exit App Builder to find out?
4. Add a Chatter Feed.
 - A. Click and drag the **Chatter Feed** component from the Standard Lightning Components palette and drop it onto the bottom-right column in the design canvas.
 - B. In the properties panel, change the Feed Type to "What I Follow".
5. Add a Filter List.
 - A. Click and drag the **List View** from the Standard Lightning Components palette and drop it onto the top-right column in the design canvas.
 - B. Configure the following properties:
 - **Object:** Bear
 - **Filter:** All
 - **Number of Records to Display:** 3
6. Activate Your App.
 - A. Click **Save** and then the **Activate** button.
 - B. Click on the Page Settings tab, then under Icon click **Change...**
 - C. Select a different icon (any other would do).
 - D. Click on the **Mobile** tab.
 - E. Click on Add page to app.
 - F. Drag the **Bears Worldwide** menu item to the top of the list.
 - G. Click on the **Lightning Experience** tab.
 - H. Click on **Ursus Park**.
 - I. Click **Add page to app**.
 - J. Drag the **Bear Worldwide** menu item to the top of the list.
 - K. Click **Save**.
 - L. Click **Back**.
7. Checkpoint.
 - A. Click **App Launcher**, then choose **Ursus Park**.
 - B. Click on the **Bears Worldwide** tab.
 - C. Make sure your application looks like this:



Note: The map lists the location of each bear. Add some more bear records with different locations around the world and you will be able to see them on the map.

8. Enable the Bears Worldwide Component to become a Quick Action.
 - A. From VS Code, open **force-app | main | default** and right-click on **aura**, or, from the menu, you can also click **View | Command Palette**. Select the following option:

>SFDX: Create Lightning Component

- B. Enter the following information:

- **Filename:** BearAction

- C. Make sure you create an Aura component and not a Lightning Web Component!
 - D. Open **BearAction.cmp** in VS Code.
 - E. Add the following code to the implements attribute of the component:

```
<aura:component
  implements="flexipage:availableForAllPageTypes,
  force:lightningQuickAction" access="global">
```

- F. Invoke the bearsWorldwide Lightning Web Component from the BearAction Aura component.

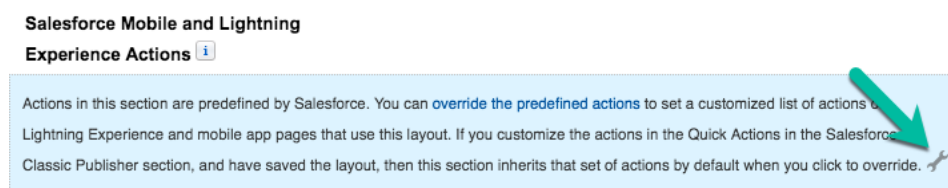
```
<aura:component
  implements="flexipage:availableForAllPageTypes,
  force:lightningQuickAction" access="global">
  <c:bearsWorldwide />
</aura:component>
```

- G. Deploy the updated code to the org. Right-click the default folder and click **SFDX: Deploy Source to Org**.
- H. Return to Lightning Experience and select **Setup | User Interface | Global Actions | Global Actions**.
- I. Click the **New Action** button.
- J. Enter the following details:

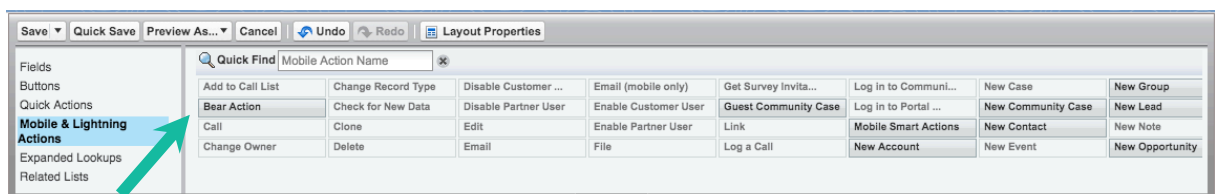
- **Action Type:** Lightning Component
- **Lightning Component:** c: BearAction
- **Height:** 450px
- **Label:** Bear Action
- **Name:** Bear_Action

- K. Click **Save**.

9. Surface the Lightning Component Quick Action.
 - A. Click **Object Manager**.
 - B. Select **Contact**.
 - C. In the Page Layouts section, click on **Contact Layout**.
 - D. Hover over the “Salesforce Mobile and Lightning Experience Actions” box and click on the wrench icon in the lower-right corner of the box.

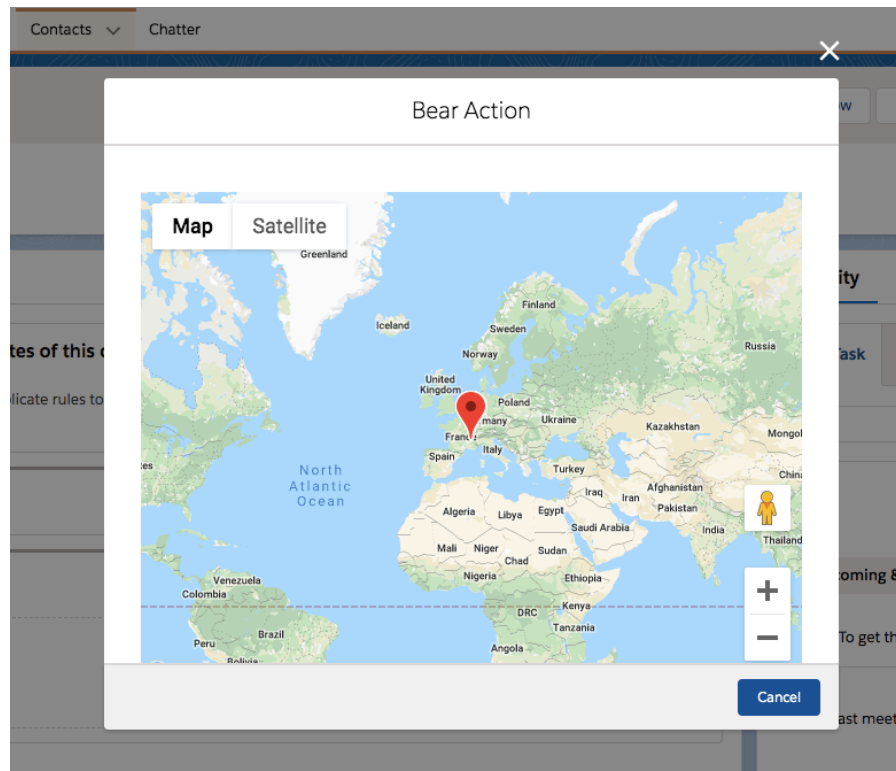


- E. In the Action Selection box, click on **Mobile & Lightning Actions**.
- F. Click on the **Bear Action** button (see below), drag and drop it onto the Salesforce mobile and Lightning Experience Actions section before Post:



- G. Click the **Save** button.

10. Checkpoint.
 - A. Open the App Launcher and switch to the **Ursus Park** app.
 - B. Select the **Contacts**.
 - C. Click on any existing contact or create a new one if needed.
 - D. Click the **Bear Action** button in the quick actions bar.



Note: The BearsWorldwide should launch in a modal popup.

E. Click **Cancel**.

Exercise 2: Surfacing Lightning Web Components in Lightning Aura Applications

Tasks:

1. Define the Application.
 2. Deploy a Lightning Web Component.
 3. Checkpoint.
 4. Create an App Template.
 5. Checkpoint.
-

1. Define the Application.
 - A. In the VS Code, open **force-app | main | default** and right-click on **aura**. Select the following option:

`>SFDX: Create Lightning App`

- B. Enter the following information:

- **Filename:** BearApp

- C. Replace the contents of BearApp.app with the following code:

```

<aura:application >
    <lightning:layout class="slds-p-around_medium"
        multipleRows="true">
        <lightning:layoutItem size="12"
            class="slds-align_absolute-center slds-m-bottom_x-small">
            <div class="slds-text-heading_large">BearApp</div>
        </lightning:layoutItem>
        <lightning:layoutItem size="12" smallDeviceSize="12"
            mediumDeviceSize="3">
            <lightning:card iconName="utility:description"
                title="Attribute Debugger" class="slds-m-horizontal_medium">
                <div class="slds-p-around_medium">
                    listView: {!v.listView} <BR />
                    markersTitle: {!v.markersTitle} <BR />
                </div>
            </lightning:card>
        <div
            class="slds-m-around_medium slds-p-around_medium">
                This page is built using an aura:application.
            </div>
        </lightning:layoutItem>
        <lightning:layoutItem size="12" smallDeviceSize="12"
            mediumDeviceSize="9">
            <!-- bearsWorldwide goes here -->
        </lightning:layoutItem>
    </lightning:layout>
</aura:application>

```

- D. Modify the <aura:application> tag to automatically load the Salesforce Lightning Design System:

```

<aura:application extends="force:slds">

```

6. Deploy a Lightning Web Component.

- A. Add <aura:attribute> tags for listView and markersTitle between the <aura:application> tags:

```

<aura:attribute name="listView" type="String"
    default="visible"/>
<aura:attribute name="markersTitle" type="String"
    default="All Bears Worldwide" />

```

- B. Find the comment that says <!-- bearsWorldwide goes here --> and replace it with an invocation of the **c:bearsWorldwide** component, passing in the listView and markersTitle attributes. Note that we are using Aura expression syntax.

```

<c:bearsWorldwide listView="{!v.listView}"
    markersTitle="{!v.markersTitle}" />

```

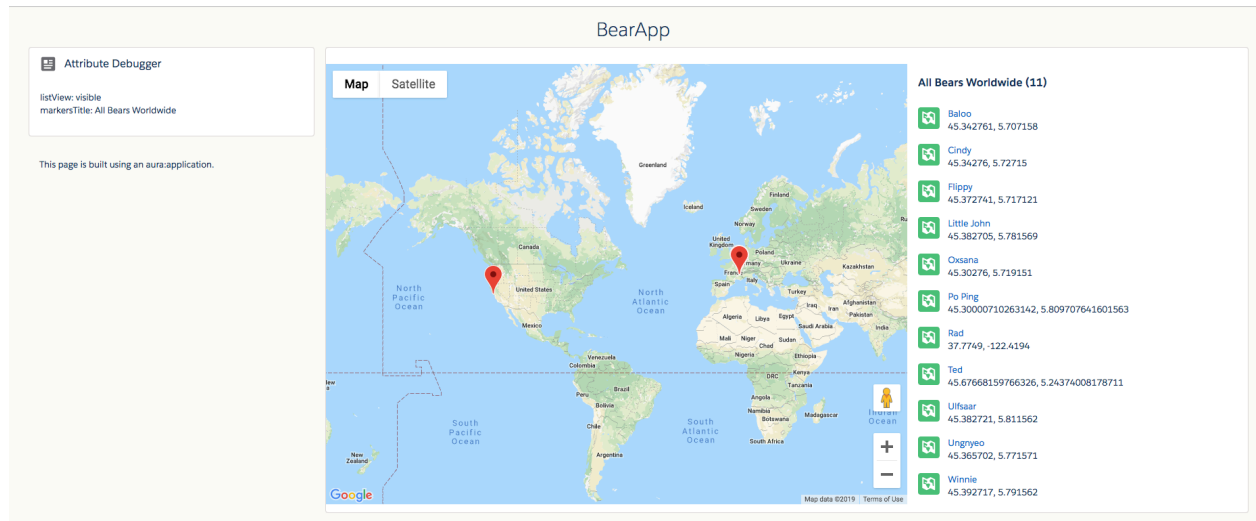
7. Checkpoint.

A. Deploy the updated code to the org. Right-click the default folder and click **SFDX: Deploy Source to Org**.

B. Open the Terminal (View | Terminal) and execute this command:

```
sfdx force:org:open --path "/c/BearApp.app"
```

C. Make sure your application looks like this:



Note: A new browser tab opens and you see the bear map on the screen. You are not running this inside Lightning Experience UI.

D. Modify the URL to suppress the list view:

```
https://<mydomain>/c/BearApp.app?listView=hidden
```

Note: When the app refreshes, the map will display without the side panel that lists the bears.

8. Create an App Template.

A. In VS Code, open **force-app | main | default** and right-click on **aura**. Select the following option:

```
SFDX: Create Lightning Component
```

B. Enter the following information:

- **Filename:** BearAppTemplate

C. Update the first line of the component to indicate this is an app template. The code should look like this:

```
<aura:component isTemplate="true" extends="aura:template">
```

- D. Inside the component, use the `<aura:set>` tag to set the HTML page title to "Bears Worldwide" and to avoid the use of a reset stylesheet.

```
<aura:set attribute="title" value="Bears Worldwide"/>
<aura:set attribute="auraResetStyle" value=""/>
```

- E. Delete the file **BearAppTemplateRenderer.js**.

Note: App template components cannot include renderers and the file will cause an error if it's not deleted.

- F. Return to **BearApp.app**.

- G. Add a template attribute to the `<aura:application>` tag that invokes the template that you just created.

```
<aura:application extends="force:slds"
    template="c:BearAppTemplate">
```

9. Checkpoint.

- A. Deploy the updated code to the org. Right-click the default folder and click **SFDX: Deploy Source to Org**.
- B. Reload the browser tab.
- C. Confirm that your browser tab's title now says "Bears Worldwide" instead of "Aura".

Exercise 3: Using Lightning in Visualforce Pages with Lightning Out

Tasks:

1. Add a new Apex method.
 2. Define an Application to Load the Components.
 3. Create the Visualforce page.
 4. Complete the TODOs to set the page up.
 5. Use JavaScript to invoke Lightning Web Components from Visualforce.
 6. Checkpoint.
-

1. Add a new Apex method.
 - A. Open **BearController.cls**.
 - B. Add the following method:

```
public Bear__c[] locationsVF {  
    get {  
        if (locationsVF == null) {  
            locationsVF = BearController.getAllBears();  
        }  
        return locationsVF;  
    }  
    private set;  
}
```

2. Define an Application to Load the Components.
 - A. In VS Code, open **force-app | main | default** and right-click on **aura**. Select the following option:

SFDX: Create Lightning App

- B. Enter the following information:

- **Filename:** vfDependency

- C. Modify the `<aura:application>` tag in the following way:

```
<aura:application extends="ltng:outApp">
```

- D. Inside the `<aura:application>` block, add an `<aura:dependency>` tag to load your **c:bearsWorldwide** component and one more for the **lightning:badge**.

```
<aura:dependency resource="c:bearsWorldwide"/>  
<aura:dependency resource="lightning:badge"/>
```

3. Create the Visualforce page.
 - A. In VS Code, open **View | Command Palette** and select the following option:

SFDX: Create Visualforce Page

B. Enter the following information:

- **Filename:** BearsWorldwideVF

C. Overwrite the contents of BearsWorldwideVF.page with the following code:

```
<apex:page controller="BearController" tabStyle="Bear__c">

    <div
style="text-align:center;margin-top:20px;margin-bottom:20px;">
        <h1 style="font-size:24px;">Lightning Web Components
in Visualforce</h1>
    </div>

    <!-- TODO #1: load the Lightning Components for
Visualforce JavaScript library. -->

    <apex:pageBlock>
        <apex:pageBlockSection>
            <apex:pageBlockSectionItem>
                <!-- TODO #2: add a div that will hold the map
once it has been created. -->
            </apex:pageBlockSectionItem>
            <apex:pageBlockTable value="{!locationsVF}"
var="location">
                <apex:column value="{!location['Name']}" />
                <apex:column
value="{!location['Location__Latitude__s']}" />
                <apex:column
value="{!location['Location__Longitude__s']}" />
            </apex:pageBlockTable>
        </apex:pageBlockSection>
    </apex:pageBlock>

    <!-- TODO #3: add a div that will hold the 'locations
found' badge once it has been created. -->

    <!-- TODO #4: add the JavaScript code to create the
cmponents -->

</apex:page>
```

4. Complete the TODOs to set the page up.

Note: The shell of the Visualforce page has been provided but a few pieces have been omitted and marked with TODOs. Follow the instructions below to complete the TODOs.

- A. TODO #1: load the Lightning Components for Visualforce JavaScript library.

```
<apex:includeLightning/>
```

- B. TODO #2: add a div that will hold the map once it has been created.

```
<div id="mapDiv" />
```

- C. TODO #3: add a div that will hold the 'locations found' badge once it has been created.

```
<div id="badgesDiv" style="margin:10px"/>
```

- D. TODO #4: copy and paste the code from
EXFiles/Exp_EX03/BearsWorldwideVF.page.txt.

5. Checkpoint.

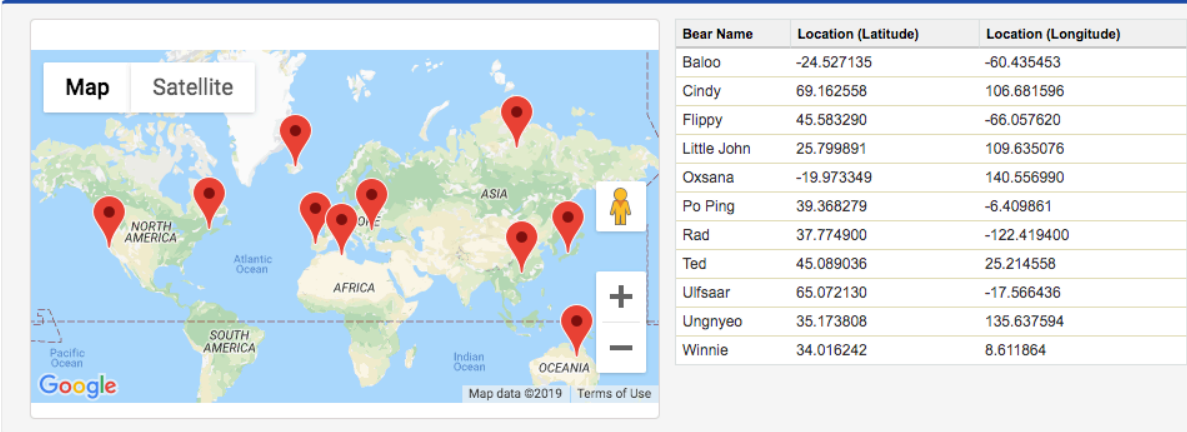
- A. Deploy the updated code to the org. Right-click the default folder and click **SFDX: Deploy Source to Org.**

- B. Open the Terminal (View | Terminal) and execute this command:

```
sfdx force:org:open --path "/apex/BearsWorldwideVF"
```

- C. Make sure your application looks like this:

Lightning Web Components in Visualforce



The screenshot displays a Visualforce page titled "Lightning Web Components in Visualforce". It features a Google Map on the left with red location pins across the world. The map includes controls for "Map" and "Satellite" views, a person icon, and zoom in/out buttons. Below the map, a badge indicates "11 locations found". To the right of the map is a table with three columns: "Bear Name", "Location (Latitude)", and "Location (Longitude)". The table lists 11 bears with their respective coordinates.

Bear Name	Location (Latitude)	Location (Longitude)
Baloo	-24.527135	-60.435453
Cindy	69.162558	106.681596
Flippy	45.583290	-66.057620
Little John	25.799891	109.635076
Oxsana	-19.973349	140.556990
Po Ping	39.368279	-6.409861
Rad	37.774900	-122.419400
Ted	45.089036	25.214558
Ulf Saar	65.072130	-17.566436
Ungnyeo	35.173808	135.637594
Winnie	34.016242	8.611864

Note: The Visualforce page loads with the bearsWorldwide component. If necessary, zoom in to see the markers on the map or update the location of the bears from the Bear record page.