

Phase 6: User Interface Development (LWC)

Overview:

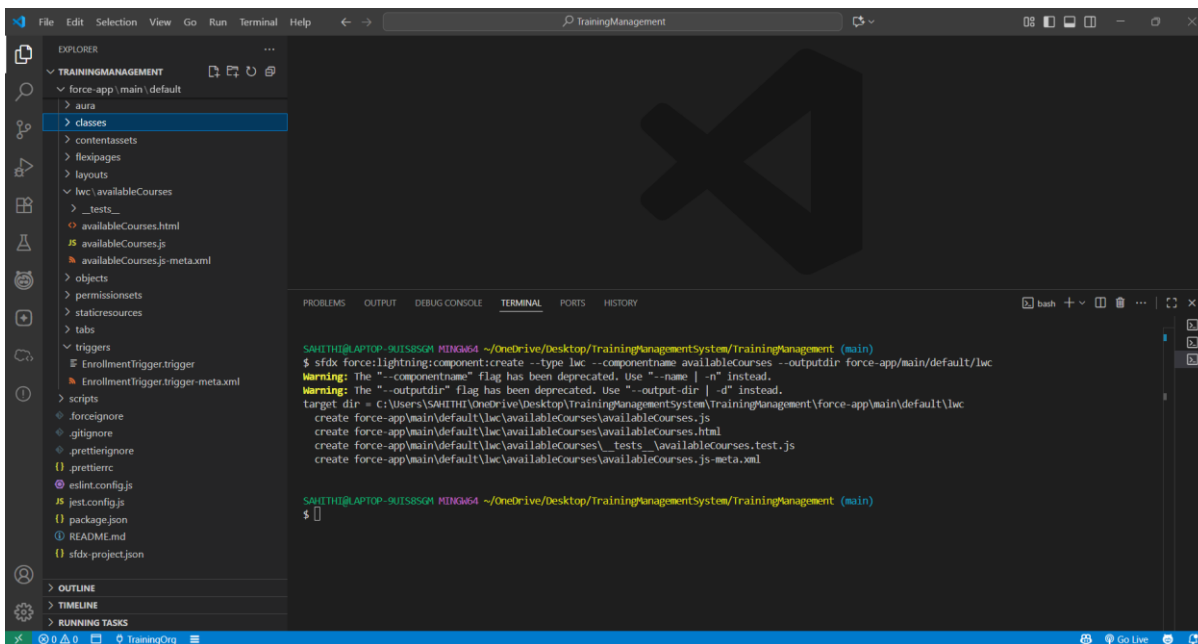
Create Lightning Web Components (LWC):

1. **Available Courses List** - Display all open course offerings
2. **Enrollment Form** - Quick enrollment component
3. **My Enrollments** - Show participant's enrollment history
4. **Course Offering Detail Card** - Display offering details with capacity indicator

Section 1:

Create Lightning Web Component - Available Courses

Step 1: Create LWC Structure

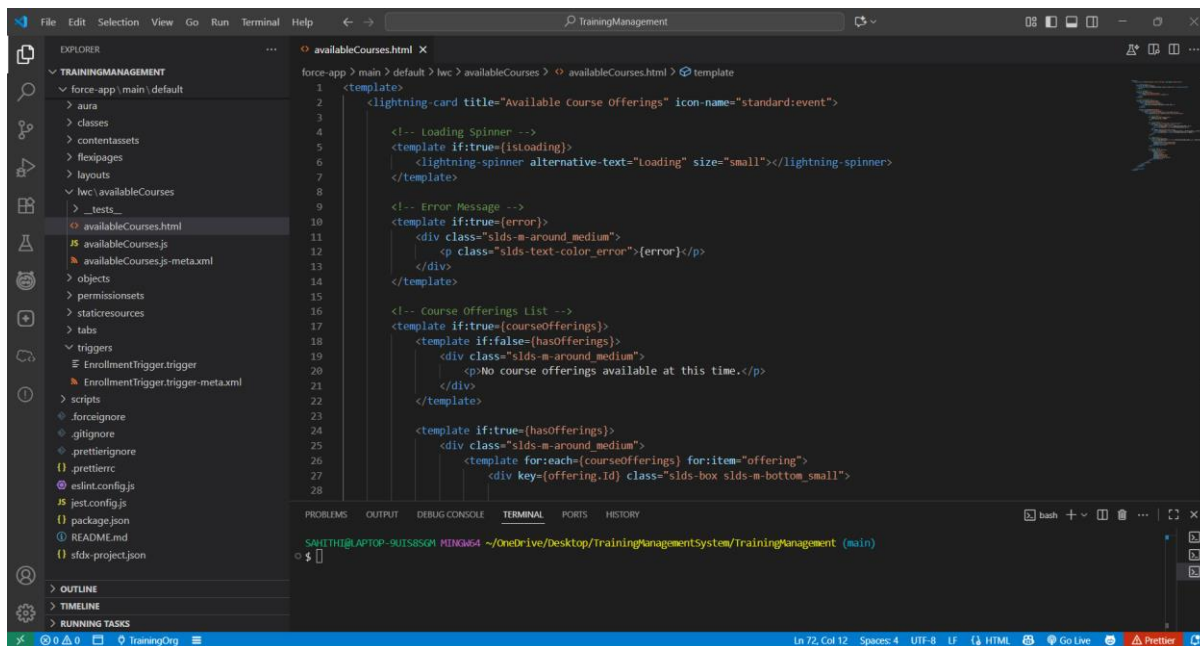


The screenshot shows the Visual Studio Code interface with the Explorer view on the left and the Terminal view at the bottom. The Explorer view shows the project structure for 'TRAININGMANAGEMENT', including folders like 'aura', 'classes', 'contentassets', 'flexpages', 'layouts', 'lwc', 'objects', 'permissionsets', 'staticresources', 'tabs', 'triggers', and 'scripts'. The 'lwc' folder is expanded, showing the 'availableCourses' component structure with files like 'availableCourses.html', 'availableCourses.js', and 'availableCourses.js-meta.xml'. The Terminal view shows the command prompt with the following commands and output:

```
SAHITHI@LAPTOP-SUITS8SGH MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
$ sfdx force:lightning:component:create --type lwc --componentname availableCourses --outputdir force-app/main/default/lwc
Warning: The "--componentname" flag has been deprecated. Use "--name | -n" instead.
Warning: The "--outputdir" flag has been deprecated. Use "--output-dir | -d" instead.
target dir = c:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\lwc
create force-app\main\default\lwc\availableCourses\availableCourses.js
create force-app\main\default\lwc\availableCourses\availableCourses.html
create force-app\main\default\lwc\availableCourses\_tests\_availableCourses.test.js
create force-app\main\default\lwc\availableCourses\availableCourses.js-meta.xml

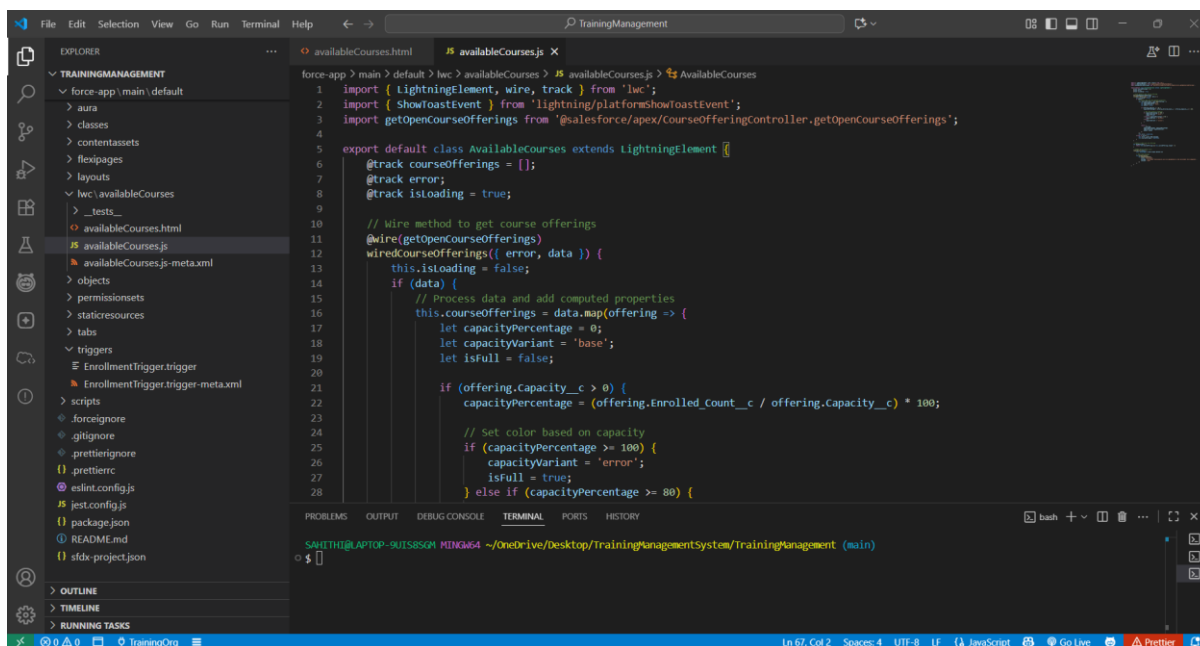
SAHITHI@LAPTOP-SUITS8SGH MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
$
```

Step 2: Edit Component HTML



```
force-app > main > default > lwc > availableCourses > availableCourses.html > template
1 <template>
2   <lightning-card title="Available Course Offerings" icon-name="standard:event">
3
4     <!-- Loading Spinner -->
5     <template if:true={isLoading}>
6       <lightning-spinner alternative-text="Loading" size="small"></lightning-spinner>
7     </template>
8
9     <!-- Error Message -->
10    <template if:true={error}>
11      <div class="slds-m-around_medium">
12        <p class="slds-text-color_error">{error}</p>
13      </div>
14    </template>
15
16    <!-- Course Offerings List -->
17    <template if:true={courseOfferings}>
18      <template if:false={hasOfferings}>
19        <div class="slds-m-around_medium">
20          <p>No course offerings available at this time.</p>
21        </div>
22      </template>
23
24      <template if:true={hasOfferings}>
25        <div class="slds-m-around_medium">
26          <template for:each={courseOfferings} for:item="offering">
27            <div key={offering.id} class="slds-box slds-m-bottom_small">
28
```

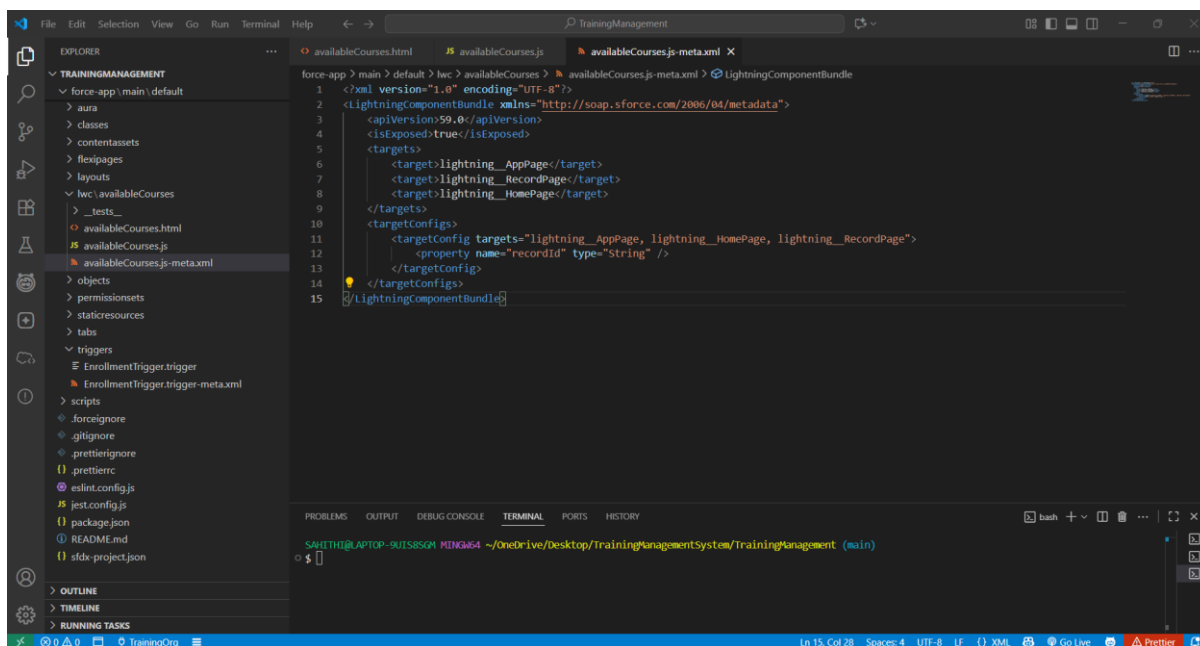
Step 3: Edit Component JavaScript



```
force-app > main > default > lwc > availableCourses > availableCourses.js > AvailableCourses
1 import { LightningElement, wire, track } from 'lwc';
2 import { ShowToastEvent } from 'lightning/platformShowToastEvent';
3 import getOpenCourseOfferings from '@salesforce/apex/CourseOfferingController.getOpenCourseOfferings';
4
5 export default class AvailableCourses extends LightningElement {
6   @track courseOfferings = [];
7   @track error;
8   @track isLoading = true;
9
10  // Wire method to get course offerings
11  @wire(getOpenCourseOfferings)
12  wiredCourseOfferings({ error, data }) {
13    this.isLoading = false;
14    if (data) {
15      // Process data and add computed properties
16      this.courseOfferings = data.map(offering => {
17        let capacityPercentage = 0;
18        let capacityVariant = 'base';
19        let isFull = false;
20
21        if (offering.Capacity__c > 0) {
22          capacityPercentage = (offering.Enrolled_Count__c / offering.Capacity__c) * 100;
23
24          // Set color based on capacity
25          if (capacityPercentage >= 100) {
26            capacityVariant = 'error';
27            isFull = true;
28          } else if (capacityPercentage >= 80) {

```

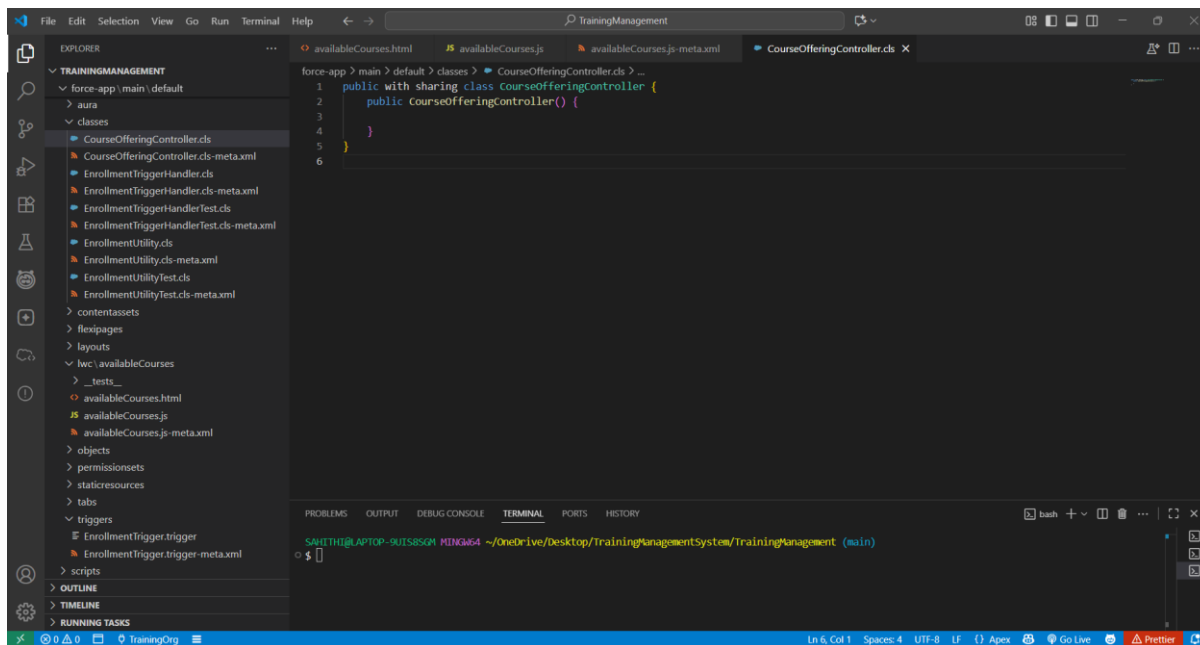
Step 4: Edit Meta XML



Section 2:

Create Apex Controller for LWC

Step 1: Create Apex Class



Step 2: Write Apex Controller Code

The screenshot shows the Visual Studio Code editor with the Apex Controller Code for `CourseOfferingController`. The code is as follows:

```
force-app > main > default > classes > CourseOfferingController.cls
1 public with sharing class CourseOfferingController {
2
3 }
4
5 @AuraEnabled
6 public static String createEnrollment(Id participantId, Id courseOfferingId) {
7     try {
8         // Create new enrollment
9         Enrollment__c enrollment = new Enrollment__c(
10             Participant__c = participantId,
11             CourseOffering__c = courseOfferingId,
12             EnrollmentDate__c = Date.today(),
13             Status__c = 'Registered'
14         );
15         insert enrollment;
16         return 'Success';
17     } catch (Exception e) {
18         throw new AuraHandledException('Error creating enrollment: ' + e.getMessage());
19     }
20 }
```

The bottom panel shows the output of the Salesforce CLI command:

```
target_dir = c:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\classes
16:34:39.527 Finished SFDX: Create Apex Class
16:35:31.183 Starting SFDX: Deploy This Source to Org
=== Deployed Source
STATE FULL NAME TYPE PROJECT PATH
Created CourseOfferingController ApexClass force-app\main\default\classes\CourseOfferingController.cls
Created CourseOfferingController ApexClass force-app\main\default\classes\CourseOfferingController.cls-meta.xml
16:35:36.350 Ended SFDX: Deploy This Source to Org
```

Section 3:

Deploy and Add LWC to Page

Step 1: Deploy LWC to Salesforce

The screenshot shows the Visual Studio Code editor with the LWC component code for `AvailableCourses`. The code is as follows:

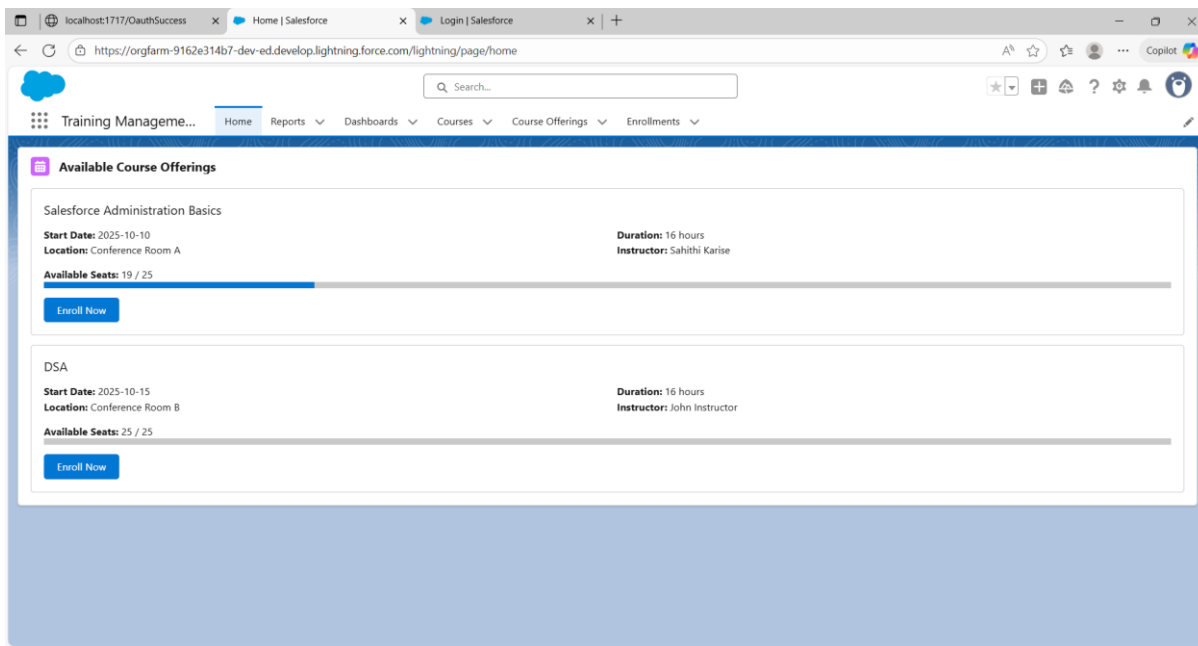
```
force-app > main > default > lwc > availableCourses > availableCourses.js
1 import { LightningElement, wire, track, api } from 'lwc';
2 import { ShowToastEvent } from 'lightning/platform/showToastEvent';
3 import { getOpenCourseOfferings } from '@salesforce/apex/CourseOfferingController.getOpenCourseOfferings';
4
5 export default class AvailableCourses extends LightningElement {
6     @api recordId;
7     @track courseOfferings = [];
8     @track error;
9     @track isLoading = true;
10
11     // Wire method to get course offerings
12     @wire(getOpenCourseOfferings)
13     wiredCourseOfferings({ error, data }) {
14         this.isLoading = false;
15         if (data) {
16             // Process data and add computed properties
17             this.courseOfferings = data.map(offering => {
18                 let capacityPercentage = 0;
19                 let capacityVariant = 'base';
20                 let isFull = false;
21
22                 if (offering.Capacity__c > 0) {
23                     capacityPercentage = (offering.EnrolledCount__c / offering.Capacity__c) * 100;
24                 }
25             });
26         }
27     }
28 }
```

The bottom panel shows the output of the Salesforce CLI command:

```
16:41:25.422 Ended SFDX: Deploy This Source to Org
16:44:04.319 Starting SFDX: Deploy This Source to Org
=== Deployed Source
STATE FULL NAME TYPE PROJECT PATH
Created availableCourses LightningComponentBundle force-app\main\default\lwc\availableCourses\availableCourses.html
Created availableCourses LightningComponentBundle force-app\main\default\lwc\availableCourses\availableCourses.js
Created availableCourses LightningComponentBundle force-app\main\default\lwc\availableCourses\availableCourses.js-meta.xml
16:44:07.211 Ended SFDX: Deploy This Source to Org
```

Controller deployed successfully

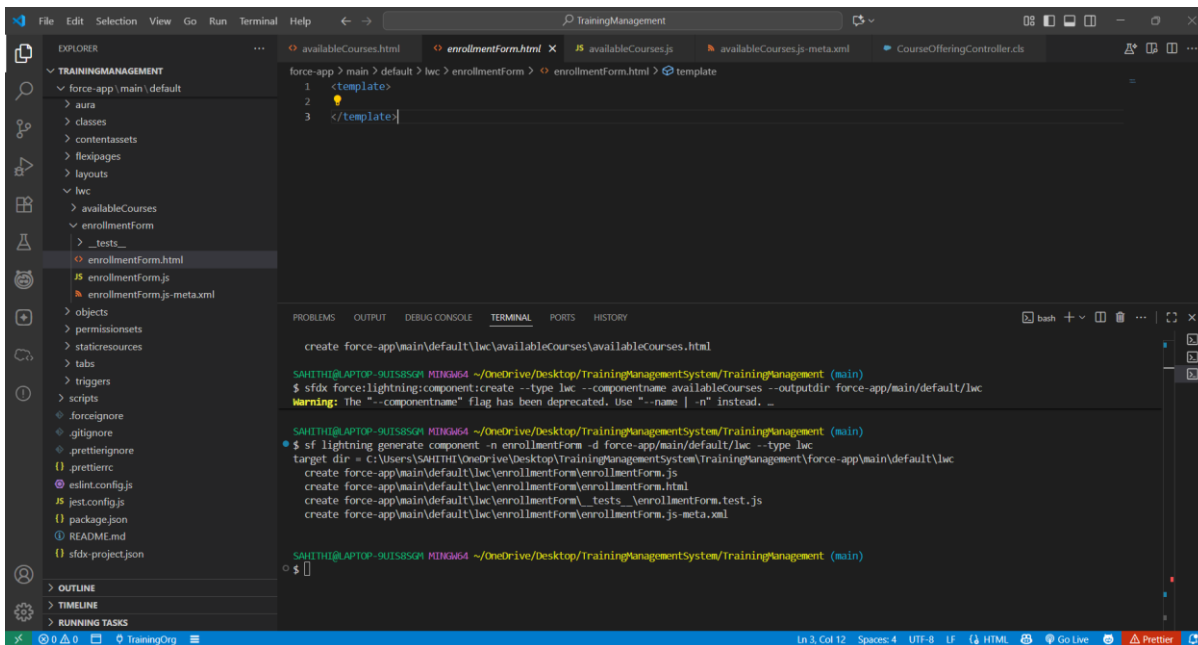
Step 2: Add Component to Home Page



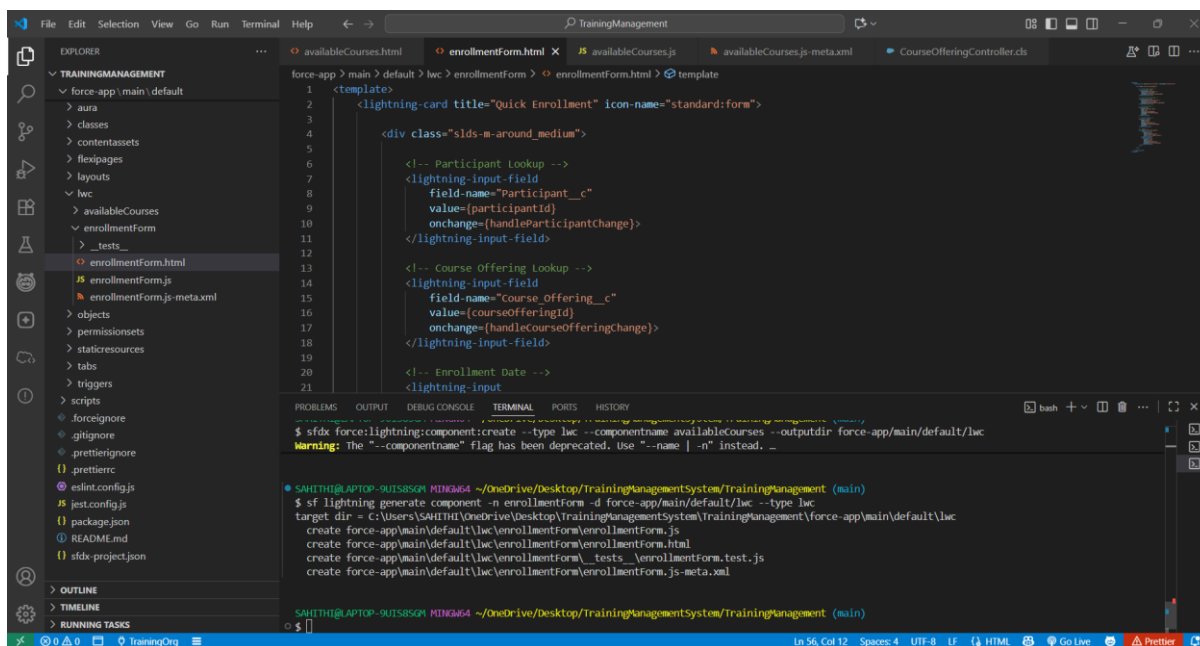
Section 4:

Create Enrollment Form Component

Step 1: Create Component



Step 2: Edit Enrollment Form HTML

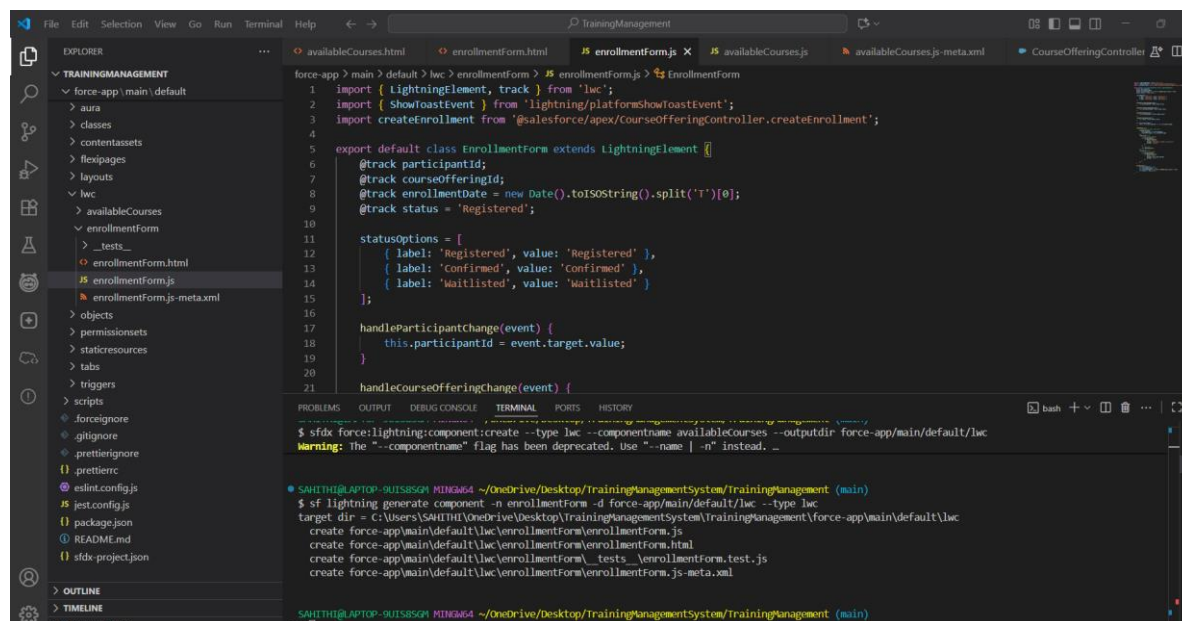


The screenshot shows the Visual Studio Code editor with the 'enrollmentForm.html' file open. The file is a Lightning component template. The Explorer on the left shows the project structure for 'TRAININGMANAGEMENT', including 'force-app/main/default' and 'enrollmentForm'. The main editor area displays the HTML template with the following content:

```
1 <template>
2   <lightning-card title="Quick Enrollment" icon-name="standard:form">
3     <div class="slds-m-around_medium">
4       <!-- Participant Lookup -->
5       <lightning-input-field
6         field-name="Participant_c"
7         value={participantId}
8         onchange={handleParticipantChange}>
9     </lightning-input-field>
10
11     <!-- Course Offering Lookup -->
12     <lightning-input-field
13       field-name="Course offering_c"
14       value={courseOfferingId}
15       onchange={handleCourseOfferingChange}>
16     </lightning-input-field>
17
18     <!-- Enrollment Date -->
19     <lightning-input
20
21
```

The TERMINAL pane at the bottom shows the command prompt output for the 'sfdx force:lightning:component:create' command, indicating that the component was successfully created.

Step 3: Edit Enrollment Form JavaScript



The screenshot shows the Visual Studio Code editor with the 'enrollmentForm.js' file open. The file contains the JavaScript logic for the enrollment form. The Explorer on the left shows the project structure for 'TRAININGMANAGEMENT', including 'force-app/main/default' and 'enrollmentForm'. The main editor area displays the JavaScript code with the following content:

```
1 import { LightningElement, track } from 'lwc';
2 import { ShowToastEvent } from 'lightning/platformShowToastEvent';
3 import createEnrollment from '@salesforce/apex/CourseOfferingController.createEnrollment';
4
5 export default class EnrollmentForm extends LightningElement {
6   @track participantId;
7   @track courseOfferingId;
8   @track enrollmentDate = new Date().toISOString().split('T')[0];
9   @track status = 'Registered';
10
11   statusOptions = [
12     { label: 'Registered', value: 'Registered' },
13     { label: 'Confirmed', value: 'Confirmed' },
14     { label: 'Waitlisted', value: 'Waitlisted' }
15   ];
16
17   handleParticipantChange(event) {
18     this.participantId = event.target.value;
19   }
20
21   handleCourseOfferingChange(event) {
22
23
```

The TERMINAL pane at the bottom shows the command prompt output for the 'sfdx force:lightning:component:create' command, indicating that the component was successfully created.

Step 4: Configure Meta XML

The screenshot shows the VS Code interface with the Explorer on the left, the main editor in the center, and the Output/Debug Console at the bottom. The Explorer shows the project structure for 'TRAININGMANAGEMENT', with 'enrollmentForm.js-meta.xml' selected. The main editor displays the content of 'enrollmentForm.js-meta.xml', which is a LightningComponentBundle configuration. The Output/Debug Console shows the command 'sfdx force:lightning:component:create' and its output, including a warning about the deprecated '--componentname' flag and the successful creation of the component and its associated files.

```
force-app > main > default > lwc > enrollmentForm > enrollmentForm.js-meta.xml > LightningComponentBundle
1 <?xml version="1.0" encoding="UTF-8"?>
2 <LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
3   <apiVersion>59.0</apiVersion>
4   <isExposed>true</isExposed>
5   <targets>
6     <target>lightning__AppPage</target>
7     <target>lightning__RecordPage</target>
8     <target>lightning__HomePage</target>
9   </targets>
10 </LightningComponentBundle>
```

```
$ sfdx force:lightning:component:create --type lwc --componentname availableCourses --outputdir force-app/main/default/lwc
Warning: The "--componentname" flag has been deprecated. Use "--name | -n" instead. _

SAHITHI@LAPTOP-9UTS8SQH MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
$ sf lightning generate component -n enrollmentForm -d force-app/main/default/lwc --type lwc
target dir = c:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\lwc
create force-app\main\default\lwc\enrollmentForm\enrollmentForm.js
create force-app\main\default\lwc\enrollmentForm\enrollmentForm.html
create force-app\main\default\lwc\enrollmentForm\_tests\_enrollmentForm.test.js
create force-app\main\default\lwc\enrollmentForm\enrollmentForm.js-meta.xml

SAHITHI@LAPTOP-9UTS8SQH MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
$
```

Step 5 : Saved and Deployed

The screenshot shows the VS Code interface with the Explorer on the left, the main editor in the center, and the Output/Debug Console at the bottom. The Explorer shows the project structure for 'TRAININGMANAGEMENT', with 'enrollmentForm.js-meta.xml' selected. The main editor displays the content of 'enrollmentForm.js-meta.xml'. The Output/Debug Console shows the command 'sfdx force:lightning:component:create' and its output, including a warning about the deprecated '--componentname' flag and the successful creation of the component and its associated files. The Output/Debug Console also shows the command 'sfdx force:lightning:deploy' and its output, including the deployment of the component and its associated files.

```
force-app > main > default > lwc > enrollmentForm > enrollmentForm.js-meta.xml > LightningComponentBundle
1 <?xml version="1.0" encoding="UTF-8"?>
2 <LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
3   <apiVersion>59.0</apiVersion>
4   <isExposed>true</isExposed>
5   <targets>
6     <target>lightning__AppPage</target>
7     <target>lightning__RecordPage</target>
8     <target>lightning__HomePage</target>
9   </targets>
10 </LightningComponentBundle>
```

```
Changed availableCourses LightningComponentBundle force-app\main\default\lwc\availableCourses\availableCourses.js-meta.xml

18:06:18.780 Ended SFDX: Deploy This Source to Org
23:43:47.632 Starting SFDX: Deploy This Source to Org

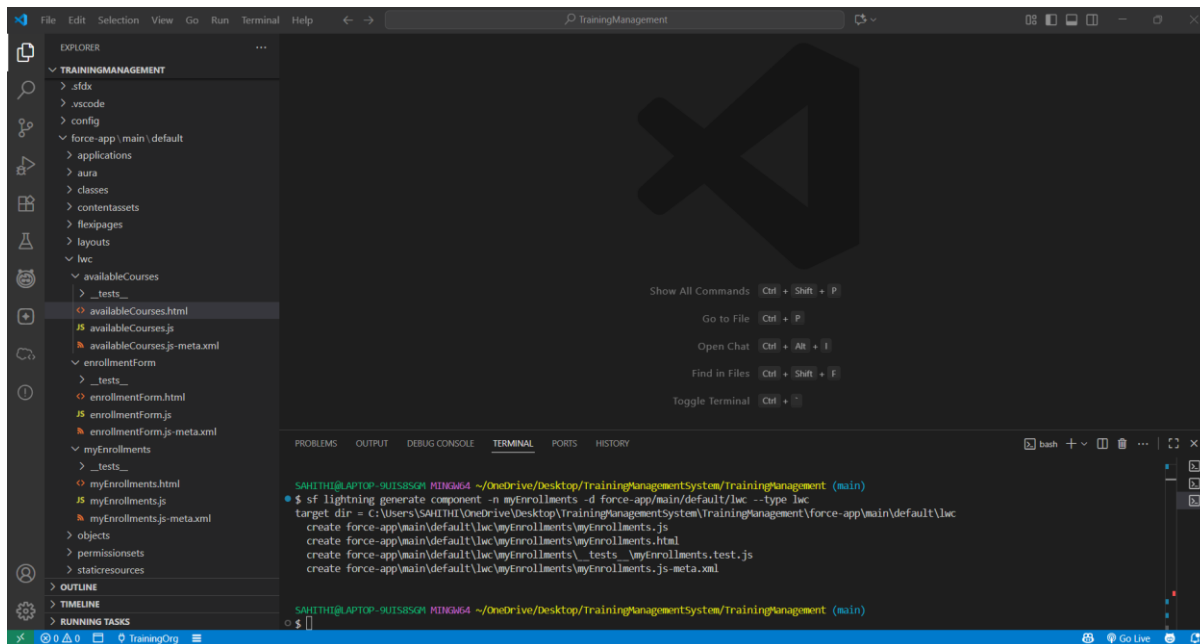
=== Deployed Source
STATE    FULL NAME                                TYPE                                PROJECT PATH
Created  enrollmentForm                           LightningComponentBundle           force-app\main\default\lwc\enrollmentForm\enrollmentForm.html
Created  enrollmentForm                           LightningComponentBundle           force-app\main\default\lwc\enrollmentForm\enrollmentForm.js
Created  enrollmentForm                           LightningComponentBundle           force-app\main\default\lwc\enrollmentForm\enrollmentForm.js-meta.xml

23:43:52.183 Ended SFDX: Deploy This Source to Org
```


Section 5:

Create My Enrollments Component

Step 1: Create Component

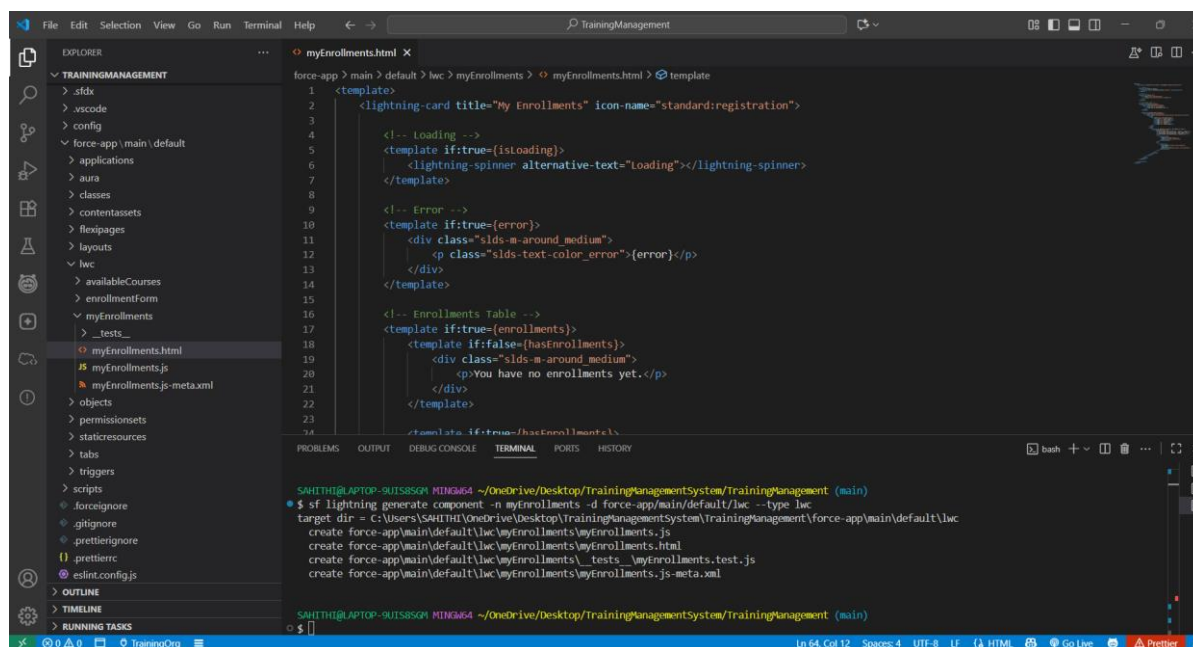


The screenshot shows the VS Code interface with the Explorer view on the left. The file tree for the 'TRAININGMANAGEMENT' project is visible, showing folders like 'force-app/main/default' and 'lwc'. The 'myEnrollments' folder is highlighted. The main editor area shows the command palette with the command 'sf lightning generate component' selected. The terminal at the bottom shows the command being executed: 'sf lightning generate component -n myEnrollments -d force-app/main/default/lwc --type lwc'. The output shows the files created: 'myEnrollments.js', 'myEnrollments.html', 'myEnrollments.js-meta.xml', 'myEnrollments.test.js', and 'myEnrollments.js-meta.xml'.

```
SAHITHI@LAPTOP-SUTSBSGQ MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
$ sf lightning generate component -n myEnrollments -d force-app/main/default/lwc --type lwc
target dir = C:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\lwc
create force-app\main\default\lwc\myEnrollments\myEnrollments.js
create force-app\main\default\lwc\myEnrollments\myEnrollments.html
create force-app\main\default\lwc\myEnrollments\_tests\_myEnrollments.test.js
create force-app\main\default\lwc\myEnrollments\myEnrollments.js-meta.xml

SAHITHI@LAPTOP-SUTSBSGQ MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
$
```

Step 2: Edit My Enrollments HTML

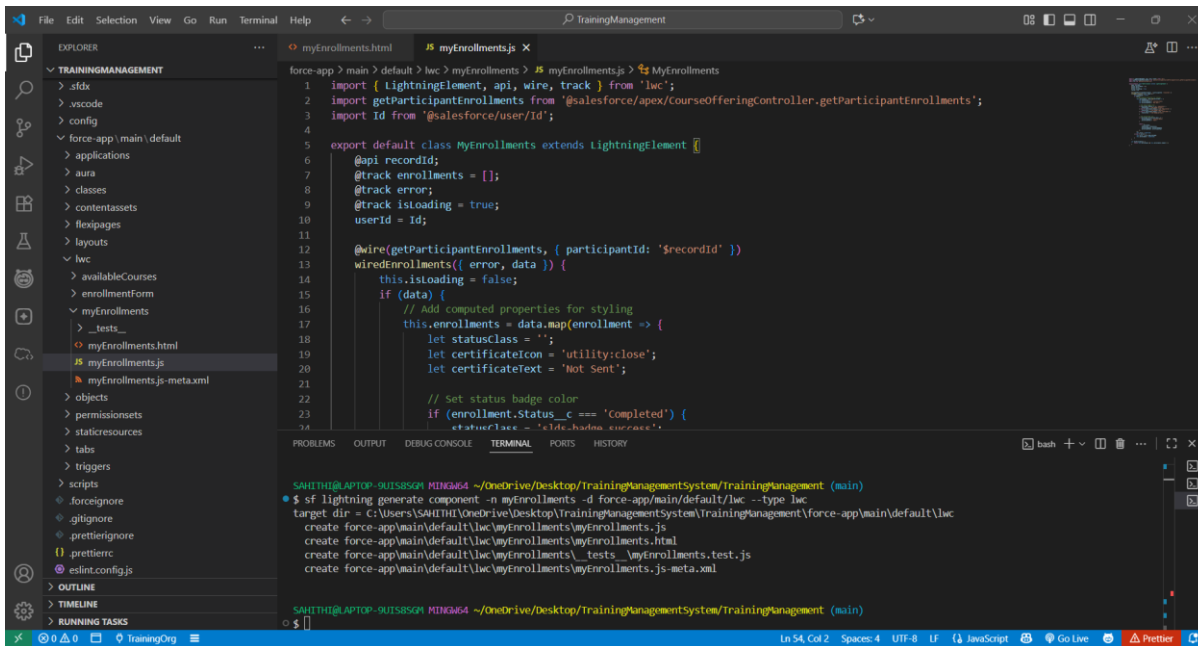


The screenshot shows the VS Code interface with the Explorer view on the left. The file tree for the 'TRAININGMANAGEMENT' project is visible, showing the 'myEnrollments' folder. The 'myEnrollments.html' file is highlighted. The main editor area shows the HTML code for the component. The terminal at the bottom shows the command being executed: 'sf lightning generate component -n myEnrollments -d force-app/main/default/lwc --type lwc'. The output shows the files created: 'myEnrollments.js', 'myEnrollments.html', 'myEnrollments.js-meta.xml', 'myEnrollments.test.js', and 'myEnrollments.js-meta.xml'.

```
SAHITHI@LAPTOP-SUTSBSGQ MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
$ sf lightning generate component -n myEnrollments -d force-app/main/default/lwc --type lwc
target dir = C:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\lwc
create force-app\main\default\lwc\myEnrollments\myEnrollments.js
create force-app\main\default\lwc\myEnrollments\myEnrollments.html
create force-app\main\default\lwc\myEnrollments\_tests\_myEnrollments.test.js
create force-app\main\default\lwc\myEnrollments\myEnrollments.js-meta.xml

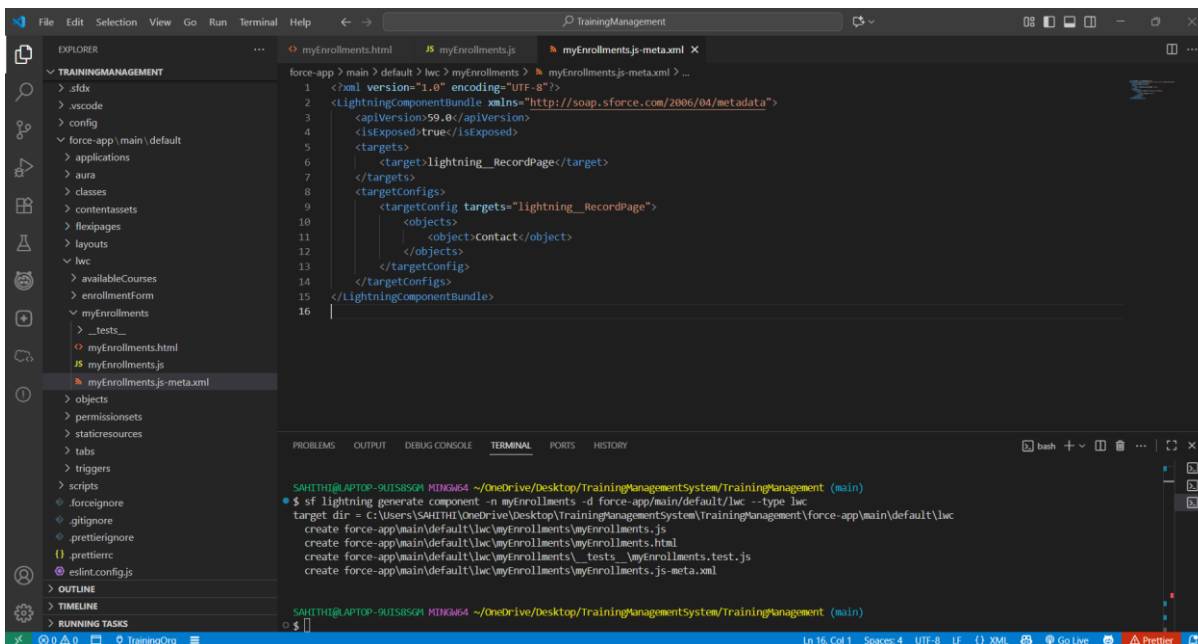
SAHITHI@LAPTOP-SUTSBSGQ MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
$
```


Step 3: Edit My Enrollments JavaScript



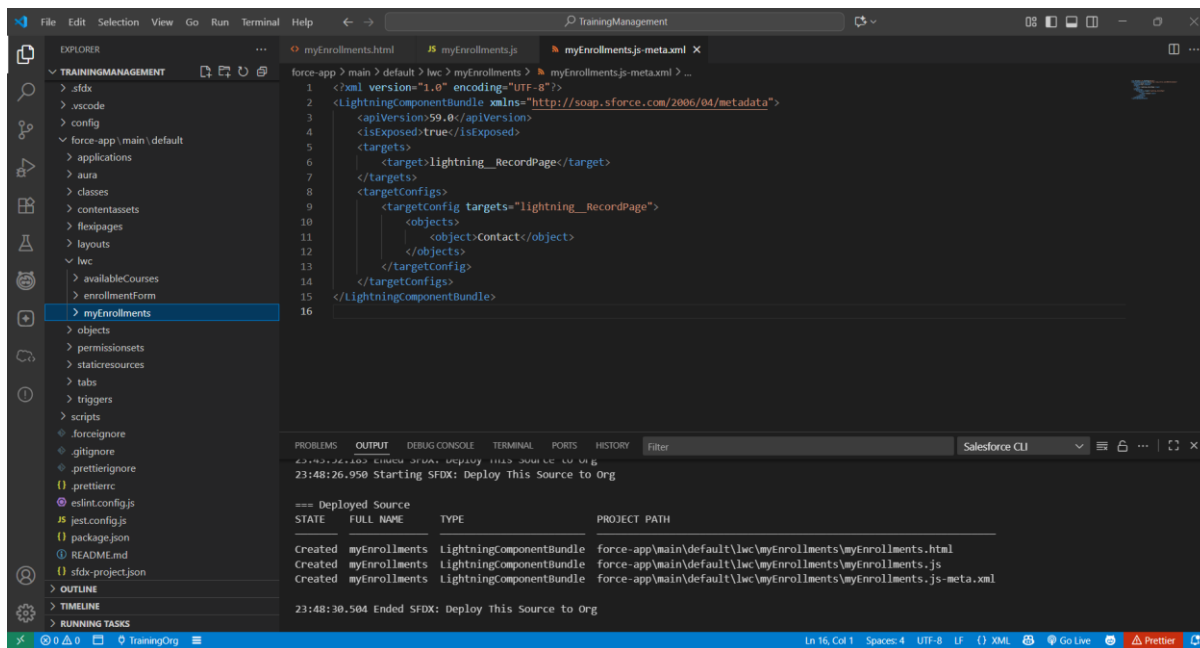
```
force-app > main > default > lwc > myEnrollments > myEnrollments.js > MyEnrollments
1 import { LightningElement, api, wire, track } from 'lwc';
2 import getParticipantEnrollments from '@salesforce/apex/CourseOfferingController.getParticipantEnrollments';
3 import Id from '@salesforce/user/Id';
4
5 export default class MyEnrollments extends LightningElement {}
6
7 @api recordId;
8 @track enrollments = [];
9 @track error;
10 @track isLoading = true;
11 userId = Id;
12
13 @wire(getParticipantEnrollments, { participantId: '$recordId' })
14 wireEnrollments({ error, data }) {
15   this.isLoading = false;
16   if (data) {
17     // Add computed properties for styling
18     this.enrollments = data.map(enrollment => {
19       let statusClass = '';
20       let certificateIcon = 'utility:close';
21       let certificateText = 'Not Sent';
22
23       // Set status badge color
24       if (enrollment.Status_c === 'Completed') {
25         certificateIcon = 'utility:checkmark';
26       }
27     });
28   }
29 }
30
31 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS HISTORY
32
33 SAHITHI@LAPTOP-9UTS8SGH MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
34 $ sf lightning generate component -n myEnrollments -d force-app/main/default/lwc --type lwc
35 target dir = c:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\lwc
36 create force-app/main/default/lwc/myEnrollments/myEnrollments.js
37 create force-app/main/default/lwc/myEnrollments/myEnrollments.html
38 create force-app/main/default/lwc/myEnrollments/_tests_myEnrollments.test.js
39 create force-app/main/default/lwc/myEnrollments/myEnrollments.js-meta.xml
40
41 SAHITHI@LAPTOP-9UTS8SGH MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
42 $
```

Step 4: Configure Meta XML



```
force-app > main > default > lwc > myEnrollments > myEnrollments.js-meta.xml > ...
1 <?xml version="1.0" encoding="UTF-8"?>
2 <LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
3   <apiVersion>59.0</apiVersion>
4   <isExposed>true</isExposed>
5   <targets>
6     <target>lightning_RecordPage</target>
7   </targets>
8   <targetConfigs>
9     <targetConfig targets="lightning_RecordPage">
10       <objects>
11         <object>Contact</object>
12       </objects>
13     </targetConfig>
14   </targetConfigs>
15 </LightningComponentBundle>
16
17 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS HISTORY
18
19 SAHITHI@LAPTOP-9UTS8SGH MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
20 $ sf lightning generate component -n myEnrollments -d force-app/main/default/lwc --type lwc
21 target dir = c:\Users\SAHITHI\OneDrive\Desktop\TrainingManagementSystem\TrainingManagement\force-app\main\default\lwc
22 create force-app/main/default/lwc/myEnrollments/myEnrollments.js
23 create force-app/main/default/lwc/myEnrollments/myEnrollments.html
24 create force-app/main/default/lwc/myEnrollments/_tests_myEnrollments.test.js
25 create force-app/main/default/lwc/myEnrollments/myEnrollments.js-meta.xml
26
27 SAHITHI@LAPTOP-9UTS8SGH MINGW64 ~/OneDrive/Desktop/TrainingManagementSystem/TrainingManagement (main)
28 $
```

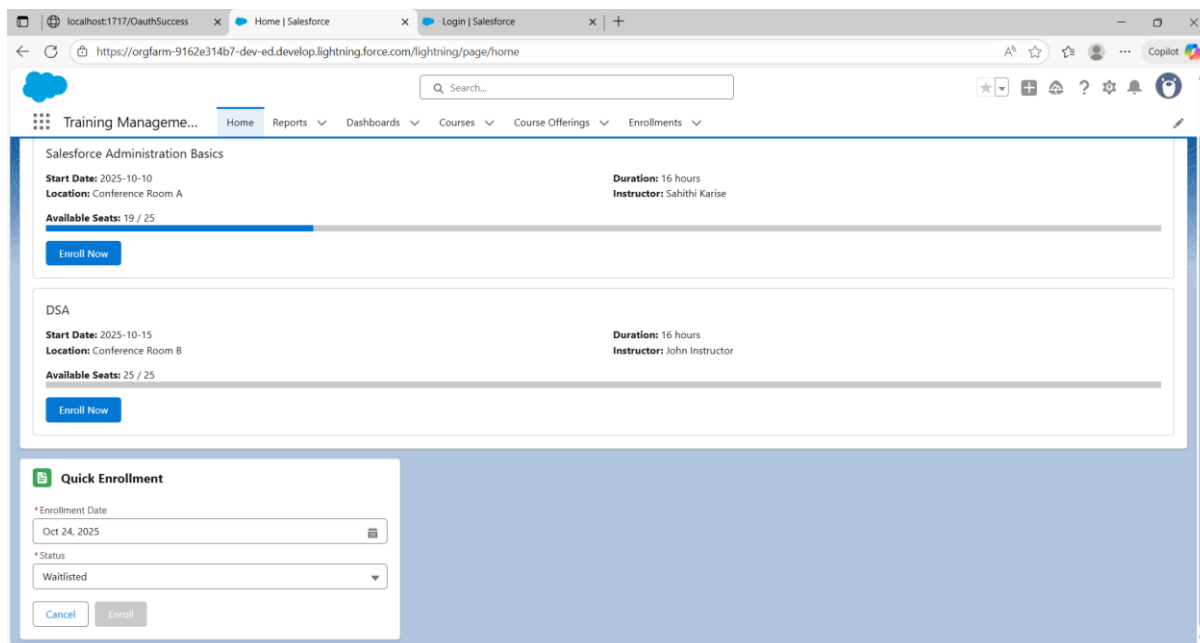
Step 5 : Save and Deploy



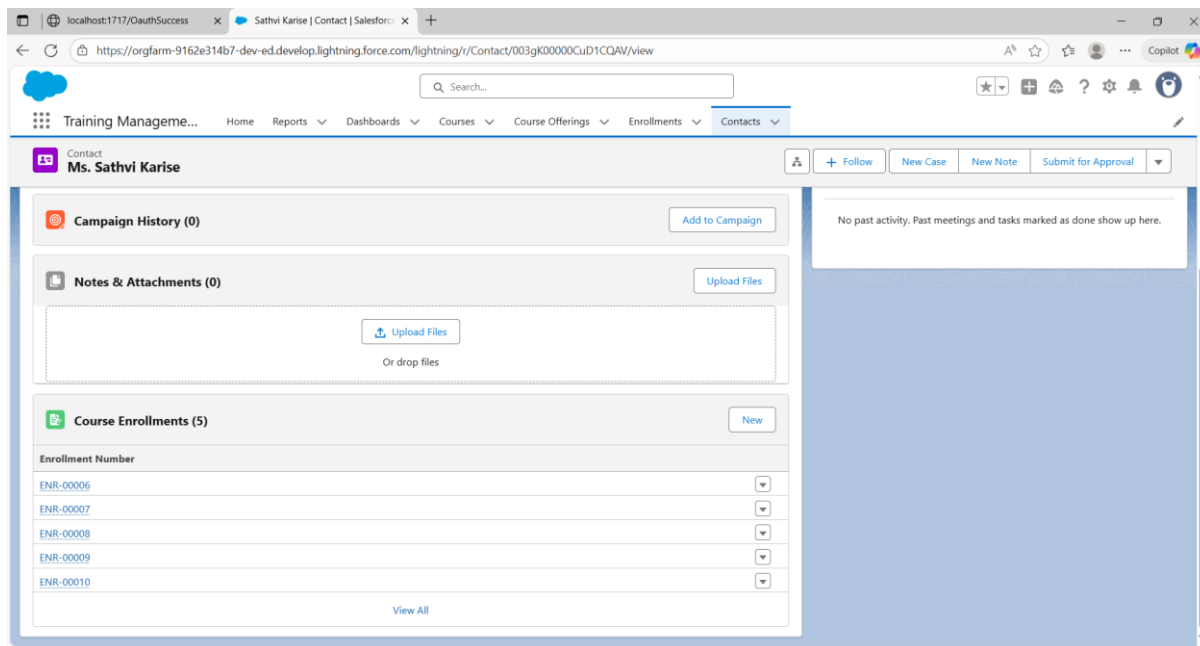
Section 6:

Add Components to Pages

Added Enrollment Form to Home Page



Added My Enrollments to Contact Page



Features Implemented:

- Real-time capacity indicators with color coding
- Interactive enrollment buttons
- Searchable participant and course lookups
- Status badges with colors
- Certificate tracking icons
- Responsive table layout

UI/UX Enhancements:

- Loading spinners
- Error handling
- Toast notifications
- Progress bars for capacity
- Conditional rendering