Real-World Example: Using Callbacks in Angular to Render a Modal Based on User Roles (`isAdmin` and `syndic`)

In this example, we'll create an Angular component that renders a modal based on the user's role (`isAdmin` or `syndic`). We'll use **callbacks** to handle the logic for determining the user's role and rendering the appropriate modal.

Scenario:

You are building an Angular application where:

1. A user can have two roles: `isAdmin` or `syndic`.
2. Depending on the role, a specific modal should be rendered.
3. The logic for determining the role and rendering the modal will be handled using **callbacks**.

Step 1: Define the User Roles and Callback Functions

We'll define two callback functions:

1. `checkIfAdmin`: Determines if the user is an admin.
2. `checkIfSyndic`: Determines if the user is a syndic.

```
// Callback to check if the user is an admin
function checkIfAdmin(user: any, callback: (isAdmin: boolean) => void): void {
  setTimeout(() => {
    const isAdmin = user.role === 'admin';
    callback(isAdmin);
  }, 1000); // Simulate async operation
}

// Callback to check if the user is a syndic
function checkIfSyndic(user: any, callback: (isSyndic: boolean) => void): void {
  setTimeout(() => {
    const isSyndic = user.role === 'syndic';
    callback(isSyndic);
  }, 1000); // Simulate async operation
}
```

Step 2: Create the Angular Component

We'll create an Angular component that uses these callbacks to determine the user's role and render the appropriate modal.

Component Code (`app.component.ts`):

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  template: `
    <div>
      <h1>User Role Modal Example</h1>
      <button (click)="determineRoleAndRenderModal()">Open Modal</button>

      <!-- Admin Modal -->
      <div *ngIf="showAdminModal" class="modal">
        <h2>Admin Modal</h2>
        <p>Welcome, Admin! You have special privileges.</p>
        <button (click)="closeModal()">Close</button>
      </div>

      <!-- Syndic Modal -->
      <div *ngIf="showSyndicModal" class="modal">
        <h2>Syndic Modal</h2>
        <p>Welcome, Syndic! You have limited privileges.</p>
        <button (click)="closeModal()">Close</button>
      </div>
    </div>
  `,
  styles: [
    `
      .modal {
        position: fixed;
        top: 50%;
        left: 50%;
        transform: translate(-50%, -50%);
        background: white;
        padding: 20px;
        border: 1px solid #ccc;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
      }
    `,
  ],
})
export class AppComponent {
  showAdminModal = false;
  showSyndicModal = false;

  // Simulated user object
  user = {
    name: 'Alice',
    role: 'admin', // Change to 'syndic' to test syndic modal
  };

  // Function to determine role and render modal
  determineRoleAndRenderModal(): void {
    // Check if the user is an admin
    checkIfAdmin(this.user, (isAdmin) => {
      if (isAdmin) {
        this.showAdminModal = true;
        return;
      }

      // If not admin, check if the user is a syndic
      checkIfSyndic(this.user, (isSyndic) => {
        if (isSyndic) {
          this.showSyndicModal = true;
        }
      });
    });
  }

  // Function to close the modal
  closeModal(): void {
    this.showAdminModal = false;
    this.showSyndicModal = false;
  }
}
```

Step 3: Explanation of the Code

1. **Callbacks for Role Checking**:
   - `checkIfAdmin` and `checkIfSyndic` simulate asynchronous operations (e.g., API calls) to determine the user's role.
   - These functions accept a `callback` parameter, which is executed once the role is determined.
2. **Component Logic**:
   - The `determineRoleAndRenderModal` method uses the callbacks to check the user's role and set the appropriate modal flag (`showAdminModal` or `showSyndicModal`).
   - If the user is an admin, the admin modal is displayed. If not, the syndic modal is displayed.
3. **Template**:
   - The template uses Angular's `*ngIf` directive to conditionally render the modals based on the `showAdminModal` and `showSyndicModal` flags.
4. **Styling**:
   - Basic CSS is used to style the modals.

Step 4: Test the Component

1. Set the `user.role` to `'admin'` and click the "Open Modal" button. The **Admin Modal** should appear.
2. Change the `user.role` to `'syndic'` and click the "Open Modal" button. The **Syndic Modal** should appear.
3. Click the "Close" button to close the modal.

Key Takeaways for Trainees:

1. **Callbacks for Asynchronous Operations**:
   - Callbacks are useful for handling asynchronous tasks like API calls or role checks.
2. **Separation of Concerns**:
   - The role-checking logic is separated into reusable callback functions.
3. **Conditional Rendering in Angular**:
   - Use `*ngIf` to conditionally render components or elements based on state.

4. **Real-World Application**:
    - This example demonstrates how callbacks can be used in Angular to handle role-based rendering.

Challenge for Trainees:

1. Add a third role (`isEditor`) and create a corresponding modal.
2. Modify the `determineRoleAndRenderModal` method to handle the new role using a callback.
3. Add a loading spinner that displays while the role is being determined.

This exercise will help trainees understand how to use callbacks effectively in Angular and handle complex role-based logic.