**MODULE - 11**

**Assignment 21: Improve Accessibility**

**Objective**

Enhance the IELTS Speaking Test platform’s interface by incorporating ARIA (Accessible Rich Internet Applications) attributes and ensuring full keyboard navigability to meet accessibility standards.

Certainly! Here is a step-by-step approach to tackle the problem of improving accessibility for the IELTS Speaking Test platform:

**Step-by-Step Instruction**

**Step 1: Analyze the Current Interface**

Audit Existing Components: Examine all interactive elements such as buttons, forms, modals, dropdowns, tabs, and any dynamic content areas for accessibility issues.

Identify Missing ARIA Attributes: Check which elements are lacking ARIA roles, states, and properties.

**Step 2: Implement ARIA Attributes**

Add ARIA Roles:

Assign appropriate roles such as role="button" for buttons, role="dialog" for modals, and role="navigation" for navigation elements.

Incorporate ARIA States and Properties:

Use aria-expanded for dropdowns or expandable sections.

Implement aria-selected for selectable elements like tabs.

Embed aria-label and aria-labelledby for labeling elements.

Implement Aria-live Regions:

Add attributes like aria-live="polite" or aria-live="assertive" for dynamically updated regions that need real-time announcements.

**Step 3: Enhance Keyboard Navigability**

Ensure Focusability:

Verify that all interactive elements are focusable using the Tab key (tabindex attribute).

Provide Visual Focus Indicators:

Use CSS to emphasize focus states (outlines, border colors, etc.)

Implement Keyboard Behavior:

Handle specific key events (onKeyDown handlers) for actions like submitting forms when pressing 'Enter'.

**Step 4: Form Accessibility**

Label Association:

Correctly link form labels with their corresponding input fields using for and id attributes.

Instruction and Error Messages:

Use aria-describedby to associate input fields with explanatory text or error messages.

**Step 5: Testing for Accessibility**

Conduct Accessibility Audits:

Use tools like Lighthouse or axe to perform a thorough accessibility audit.

Identify and resolve any flagged issues in the audit report.

Verify Keyboard Navigation:

Manually test the interface using only the keyboard.

Ensure logical and accessible Tab order cycling through interactive elements.

**Step 6: Real-Time Feature Testing**

Verify ARIA-live Regions:

Ensure updates in dynamic regions are announced to screen readers with proper timing and frequency.

**Step 7: Documentation and Evidence**

Prepare Updated Files:

Update the React components with implemented ARIA attributes and keyboard navigability.

Compile Testing Evidence:

Document Lighthouse or axe accessibility reports.

Capture screenshots of audit results demonstrating resolved issues.

Explanation of Implemented Features:

Provide a brief explanation detailing accessibility improvements.

**Deliverables:**

Updated interface with ARIA attributes.

Fully keyboard-navigable UI with focus indicators.

Evidence of accessibility testing and resolution.

**Evaluation Criteria:**

Meaningful use of ARIA attributes (40%)

Keyboard navigability (30%)

Compliance demonstrated through accessibility audits (20%)

Submission completeness (10%)