**Accessibility Strategy for Teams**

**Building Accessibility into the Development Workflow**

* **Shift Left Approach:** Integrate accessibility considerations from the earliest design and planning phases, not as an afterthought.
* **Design Phase:**
  + Include accessibility requirements in design specs.
  + Use accessible design patterns, color contrast checks, and prototype testing with assistive tech.
* **Development Phase:**
  + Use semantic HTML and ARIA best practices by default.
  + Incorporate automated accessibility testing tools (like Axe, Lighthouse) into CI/CD pipelines to catch regressions early.
  + Encourage pair programming or code reviews with accessibility checklists.
* **Testing Phase:**
  + Perform manual accessibility testing (keyboard navigation, screen reader usage) alongside functional testing.
  + Include people with disabilities in usability testing when possible.
* **Deployment & Maintenance:**
  + Establish monitoring for accessibility issues post-deployment.
  + Plan regular audits and updates as technologies and guidelines evolve.

**Roles and Responsibilities**

**1. Designers**

* Create accessible wireframes and mockups with color contrast, readable fonts, and clear focus indicators.
* Use accessibility tools in design software (e.g., Figma Accessibility plugins).
* Provide detailed design specs addressing accessibility (e.g., alt text guidelines, form label requirements).

**2. Developers**

* Implement semantic HTML and ARIA properly.
* Write accessible CSS and JavaScript to ensure keyboard navigation and screen reader compatibility.
* Integrate automated accessibility tools into development workflows.
* Fix accessibility defects and ensure new code meets standards.

**3. Quality Assurance (QA) Engineers**

* Include accessibility testing in test plans.
* Use automated tools and manual testing methods (keyboard, screen readers).
* Report accessibility issues with clear reproduction steps and WCAG references.
* Verify fixes before release.

**4. Product Owners / Project Managers**

* Prioritize accessibility as a non-negotiable feature requirement.
* Allocate time and resources for accessibility tasks in sprint planning.
* Ensure team training and awareness on accessibility topics.

**Accessibility Documentation and Audits**

**Documentation**

* Maintain an internal **Accessibility Guidelines Document** outlining standards, patterns, and tools approved by the organization.
* Create reusable **Accessible Component Libraries** with documented usage and known limitations.
* Maintain detailed **Accessibility Test Reports** for releases including issue logs and remediation status.
* Keep a **Known Issues Log** for accessibility defects that are deferred with justification and timelines.

**Audits**

* Schedule regular accessibility audits (quarterly, bi-annually) combining automated scans and manual testing.
* Use audits to identify regressions, new issues, and areas needing improvement.
* Incorporate audit findings into sprint backlogs with prioritization based on severity and impact.
* Include stakeholder reviews and user feedback sessions, especially involving people with disabilities.

**Bonus: Accessibility Training and Culture**

* Conduct regular team workshops and training sessions on accessibility basics and advanced topics.
* Encourage sharing accessibility success stories and lessons learned to build empathy and awareness.
* Promote an inclusive mindset where accessibility is part of “quality” and “user experience” rather than a separate checkbox.

**Role-Specific Accessibility Checklists**

**Designers Checklist**

* Use sufficient color contrast (≥4.5:1 for text, ≥3:1 for large text).
* Provide clear focus indicators in designs.
* Use readable fonts and appropriate sizes.
* Include descriptive text alternatives for images/icons.
* Design accessible form fields with clear labels and error messages.
* Ensure interactive elements have large enough touch targets (at least 44x44 px).
* Provide skip navigation and landmarks in layout design.
* Use accessible components or note accessibility requirements in design specs.
* Test prototypes with screen readers or accessibility plugins.

**Developers Checklist**

* Use semantic HTML elements instead of divs/spans for UI components.
* Implement ARIA roles and properties only where necessary and correctly.
* Ensure keyboard accessibility: all interactive elements reachable and operable via keyboard.
* Test focus order and manage focus especially for modals/dialogs.
* Provide alt text for images or use aria-hidden="true" when decorative.
* Validate color contrast using automated tools or CSS testing.
* Handle dynamic content updates with ARIA live regions.
* Avoid keyboard traps and test escape mechanisms for overlays.
* Run automated accessibility checks on every build (e.g., Axe, Lighthouse).

**QA Engineers Checklist**

* Include accessibility test cases in test plans.
* Use automated tools (Axe, WAVE, Lighthouse) to scan tested pages.
* Perform manual keyboard navigation tests, including tab, shift+tab, arrow keys.
* Test with screen readers (NVDA, VoiceOver) on multiple browsers.
* Check form validation messages and error handling for accessibility.
* Verify ARIA attributes and dynamic content announcements.
* Report accessibility issues with clear steps, screenshots, and WCAG references.
* Verify fixes and regression test accessibility defects before releases.

**2. Sample Accessibility Guidelines Document Outline**

**1. Introduction**

* Purpose and scope
* Accessibility standards referenced (WCAG 2.1 Level AA)

**2. Design Guidelines**

* Color contrast requirements
* Typography and readability
* Focus indicators and keyboard focus styles
* Use of icons and images with text alternatives
* Accessible form design and error handling

**3. Development Guidelines**

* Semantic HTML usage
* ARIA roles and best practices
* Keyboard accessibility patterns
* Responsive and mobile-friendly accessibility
* Handling dynamic content updates
* Media accessibility (captions, transcripts)

**4. Testing Guidelines**

* Automated tool integration and usage
* Manual testing procedures (keyboard, screen reader)
* Test case examples for accessibility
* Reporting and tracking issues

**5. Tools and Resources**

* List of recommended tools (Axe, Lighthouse, WAVE, NVDA, VoiceOver)
* Training materials and reference links

**6. Roles and Responsibilities**

* Summary of team roles and accessibility duties

**7. Continuous Improvement**

* Audit schedule
* Feedback and user testing inclusion
* Updating guidelines and training

**3. Accessibility Integration Workflow (Description for Diagram)**

1. **Requirement Gathering**
   * Include accessibility criteria in user stories/requirements.
2. **Design**
   * Create accessible wireframes and prototypes.
   * Run early accessibility checks on designs.
3. **Development**
   * Implement using semantic HTML and ARIA where needed.
   * Conduct developer self-tests with automated tools.
   * Peer code reviews with accessibility checklist.
4. **Quality Assurance**
   * Automated scans of builds using Axe/Lighthouse.
   * Manual keyboard and screen reader testing.
   * Log and report accessibility issues.
5. **Release**
   * Accessibility approval before deployment.
   * Monitor feedback from users and tools post-launch.
6. **Maintenance**
   * Regular audits and re-testing.
   * Continuous training and process improvements.