IST4035 MID SEMESTRE EXAM

SECTION A: THEORY

1. Semantic HTML5, landmarks, and alt text policy [5 marks]

Semantic outline: Rather of simply utilizing

and, I would design the "Student Opportunity Board" with appropriate semantic elements.  
  
The site title and logo would be displayed in a at the top of the page.  
  
For assistive technology to identify it as a menu, the global navigation would be placed inside a landmark.  
  
would contain the main content, which would be a list of opportunities. Since each opportunity represents self-contained material, it can be a or .  
  
I would use paragraphs or for sponsor, location, and date, and headings (

**for each opportunity title) inside each card.**

Alternative text:  
  
Since the decorative image doesn't provide any useful information, screen readers should avoid it and it should have alt="".  
  
To ensure that visually impaired users receive the same information, the instructive image (such as a sponsor logo that identifies the organization) needs a meaningful alt, such as alt="Google Student Internship logo".  
  
Use of ARIA: I would add aria-expanded="true/false" to a collapsible section or a "more details" option as a minimum ARIA attribute. Screen readers cannot readily see that expanded state from HTML alone.  
Semantic HTML makes the code clearer and easier to maintain, and all of these choices facilitate easy navigation by assistive tools (such as screen readers).

1. Responsive layout strategy: Flexbox or Grid [5 marks]

Because the opportunity cards provide a predictable grid structure that requires both rows and columns to line up properly on wide screens, I would use CSS Grid.  
While Grid offers greater control over multi-column wrapping, Flexbox is excellent for one-dimensional layouts.  
  
Mobile-first strategy:  
For small screens, begin with a single column layout (one card per row), then transition to multiple columns at a predetermined breakpoint.

:root {

--space-md: 1rem;

--space-lg: 2rem;

--color-bg: #f9fafb;

--color-accent: #007bff;

}

.opportunities {

display: grid;

grid-template-columns: 1fr;

gap: var(--space-md);

padding: var(--space-lg);

background: var(--color-bg);

}

/\* breakpoint for tablets/desktops \*/

@media (min-width: 768px) {

.opportunities {

grid-template-columns: repeat(auto-fill, minmax(250px, 1fr));

}

}

When CSS variables are used, selectors don't need to be rewritten in order to change the color scheme, spacing, or theme later on. As a result, design updates happen considerably more quickly.

1. Event handling design without per-node listeners [5 marks]

It is inefficient to give each "Save" or "Remove" button its own click handler.  
Because each button has a listener, it uses more memory. If the list is re-rendered, the listeners can be lost or duplicated, which would result in DOM churn.  
  
Delegated event approach: For instance, I would affix a single listener to the container containing all of the cards.

document.querySelector('.opportunities').addEventListener('click', (e) => {

if (e.target.matches('.save-btn')) {

const card = e.target.closest('.card');

console.log('Saved:', card.dataset.id);

}

if (e.target.matches('.remove-btn')) {

const card = e.target.closest('.card');

console.log('Removed:', card.dataset.id);

}

});

I could determine which opportunity was clicked by looking at the data-id field on each card.  
There is no need to reattach listeners with this strategy, which remains stable even if the list changes or filters dynamically.

Q4. Modern web paradigms and workflow [5 marks]

i. Mobile-first and progressive enhancement: The base layout is compatible with all devices when mobile-first CSS is used first and features are added gradually. Even on slower networks, it guarantees a quick first render and lessens the need for overrides.  
  
ii. Tools and workflow: To write clean code on the first day, I would use Visual Studio Code with add-ons like Prettier and ESLint. Additionally, I would verify accessibility and performance assessments using Chrome DevTools. When Git is used with appropriate commits, the project history remains clear and facilitates collaboration.

iii. Risk of designing exclusively for desktops: If I only create for desktops, mobile consumers can have unreadable text, slow loading times, or broken layouts. Given that the majority of people reach the web initially through their phones, it may result in poor accessibility and a negative first impression.