Khaula Molapo

20240001

Web and Mobile Application Development

Week: 5

1.

A responsive design ensures that a website or mobile application adjusts properly to different screen sizes and devices. For example, consider an online learning dashboard used by university students. The dashboard displays grades, announcements, attendance records, and assignment submissions. When accessed on a desktop computer, the layout shows multiple panels side by side. The navigation menu appears on the left, the main content is in the center, and notifications are displayed on the right side.

When the same dashboard is opened on a smartphone, the layout automatically changes. The side menu becomes a small menu button at the top, and all panels stack vertically. Text becomes slightly larger, and images resize to fit the screen width. Buttons are spaced properly for touch interaction. This improves usability and ensures students can access their information easily from any device. Responsive design increases accessibility, improves user experience, and ensures consistency across platforms.

2.

Media queries are a feature in CSS used to apply different styles depending on the screen size, resolution, or device type. They are a key part of responsive web design. Instead of building separate websites for desktop and mobile devices, developers use media queries to adjust the layout within the same website.

For example, a media query can detect if the screen width is below 768 pixels. If this condition is true, the layout changes to suit tablets or smartphones. Font sizes may increase for readability, navigation menus may stack vertically, and images may resize automatically. Media queries help websites look clean and organized on all devices. They improve flexibility, reduce development time, and ensure users have a smooth experience regardless of the device they use.

3.

In this exercise, I created a simple responsive web layout using HTML and CSS. The page included a header, navigation bar, main content section, sidebar, and footer. On larger screens, the sidebar appeared next to the main content, and the navigation links were displayed horizontally.

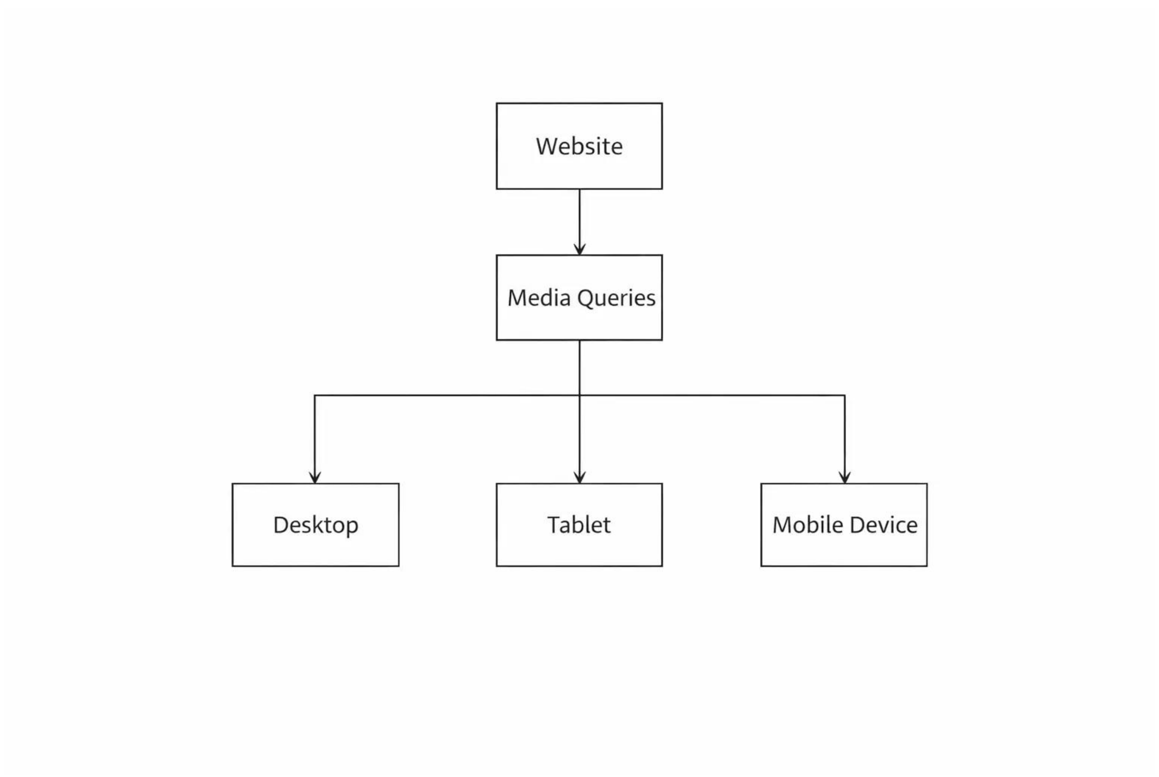
I then added CSS media queries to adjust the layout for smaller screens. When the screen width decreased, the sidebar moved below the main content, and the navigation links stacked vertically. I also used flexible units such as percentages and the flexbox layout model to allow elements to adjust naturally. This activity helped me understand how layout structure affects user experience. It also showed how careful testing is needed to ensure the design works well on different devices.

4.

In Draw.io, I created a simple responsive design flowchart using rectangles and arrows. The flowchart begins with the step 'User Opens Website.' The next step is 'System Detects Screen Size.'

From this point, the flow splits into two decision paths. If the screen size is large, the system loads the desktop layout. If the screen size is small, the system loads the mobile layout. Both paths then connect to the step 'Apply Media Queries,' which adjusts styles, layout, and content display. The final step is 'Display Optimized Interface.'

This flowchart clearly explains how responsive design works. It shows the logical process behind layout adjustment and helps visualize how different devices receive the appropriate version of the website.


Website
Media Queries
Desktop
Tablet
Mobile Device