**Course 1: Introduction To Front-End Development**

**Modul 2**

**Topic 1: Intro to UI Frameworks and Libraries**

### **Video 1: 1. Building Applications and Including Dependencies**

* Developers can build websites from scratch or use existing code like libraries and frameworks.
* Applications interact with APIs provided by libraries and frameworks.
* Libraries and frameworks are called **dependencies** because applications depend on them.
* Without including these dependencies, an application cannot execute API functions.

**2. Including Libraries in HTML**

* **CSS Library**:
  + Include using the <link> tag in the <head> section.
  + Example: Bootstrap CSS library.
* **JavaScript Library**:
  + Include using the <script> tag in the <body> section.
  + Example: Bootstrap JavaScript library for enhanced functionality like drop-downs and tooltips.

**3. Example with Bootstrap**

* Add a button to a webpage using Bootstrap:
  + Use the <button> element with type, class="btn btn-primary", and description like "Click this button."
  + Demonstrates Bootstrap's primary blue-colored button.

**4. Dependency Trees**

* Dependencies can have their own dependencies, forming a **dependency tree**.
* Managing large projects with many dependencies and their trees can be time-consuming.

**5. Package Managers**

* A **package manager** automatically downloads, installs, and configures dependencies.
  + Examples: Node Package Manager (**npm**).
* Benefits:
  + Handles dependency trees.
  + Ensures the same versions are used by a team.
  + Allows specification of dependency versions.
* Can also publish custom packages.

**6. Bundling Tools**

* A **bundler** combines multiple dependencies into a single or multiple files for efficiency.
  + Examples: **Gulp**, **Webpack**.
* Purpose:
  + Simplifies including dependencies in HTML.
  + Splits large bundles into smaller ones if needed.

**7. Key Concepts Explained**

* **Dependencies**: Libraries and frameworks required by your application.
* **Package Manager**: Tool for managing dependencies (e.g., npm).
* **Bundler**: Tool for combining dependencies into bundles for use in the application.

### **Video 2: Introduction to Responsive Design**

### **1. Responsive Design Overview**

* **Responsive Design**: Enables web pages to automatically adjust their layout based on the device's screen size, providing the best user experience.
* Essential for handling diverse screen sizes, resolutions, and high-definition displays.

**2. Core Components of Responsive Design**

Responsive design relies on three key techniques:

1. **Flexible Grids**:
   * Consist of columns, gutters (space between columns), and margins (space between content and screen edges).
   * Use percentage values instead of pixels for element sizes, allowing adaptability to screen size.
2. **Fluid Images**:
   * Set max-width to 100% in CSS for images.
   * Allows images to scale down if their container becomes narrower but prevents them from growing larger than their original size.
3. **Media Queries**:
   * CSS rules that apply conditionally based on display size, orientation, or aspect ratio.
   * Example: Change background color for screens less than or equal to 700 pixels wide.

**3. Screen Resolution and Pixels**

* **Screen Resolution**: Number of horizontal and vertical pixels (e.g., 1920x1080).
* **High-Resolution Screens**:
  + Group multiple physical pixels into one logical pixel for smoother visuals.
  + Common in smartphones, improving text and image clarity.

**4. Breakpoints in Responsive Design**

* **Breakpoint**: The pixel value at which a website's content and layout adapt to deliver the best user experience.
* Used with grids to define how the layout changes based on screen size.

**5. Types of Grids in Responsive Design**

1. **Fixed Grids**:
   * Fixed-width columns with flexible margins.
   * Content width remains constant within a breakpoint range.
2. **Fluid or Full-Width Grids**:
   * Flexible columns that grow or shrink based on screen size.
   * Fixed gutters and side margins.
3. **Hybrid Grids**:
   * Combine fixed-width and fluid-width components.
   * Offer a mix of flexibility and stability in layout.

**6. Importance of Responsive Design**

* Ensures websites are displayed correctly across devices.
* Handles challenges of various screen resolutions and high-definition displays.
* Provides the foundation for frameworks like **Bootstrap** to create mobile-first, responsive sites.