Lesson Plan: Introduction to React and Next.js for Building Websites

Total Duration: 2 Hours

1) Setup and Introduction (20 minutes)

- Overview of React and Next.js (5 min)

  React: React is a JavaScript library used for building user interfaces, particularly for single-page applications. It's maintained by Facebook and a community of individual developers and companies. React allows developers to create large web applications that can change data, without reloading the page. Its key feature is the ability to build components, which are small, reusable pieces of code that dictate how a portion of the UI should appear and behave. This modular approach makes code more manageable and scalable. React's significance in web development lies in its flexibility, efficiency, and widespread community support, making it one of the most popular tools for front-end development.

  Next.js: Next.js is a React-based framework that enables functionality such as server-side rendering and generating static websites for React-based web applications. This is especially useful for multi-page applications. A traditional React app would send a blank page to the browser and then use JavaScript to populate content, which can be less efficient and potentially harmful to SEO. Next.js allows the initial HTML to be rendered by the server, which can result in faster page load times, better SEO, and a more robust user experience. This makes Next.js an essential tool for developers looking to enhance and optimize their React applications, especially when dealing with complex, multi-page websites.

- Installation and Project Setup (15 min):

- Guide through installing Node.js, React, and Next.js.

1. Installing Node.js

Node.js is required for React and Next.js development. It comes with npm (Node Package Manager) which is used to install packages.

For Windows and macOS:

- Visit the Node.js website: https://nodejs.org/en/download/
- Download the installer for your operating system (Windows or macOS).
- Run the installer and follow the prompts to install Node.js and npm.
- After installation, verify the installation by opening a terminal (Command Prompt for Windows, Terminal for macOS) and running:
  - node -v (This should show the installed version of Node.js)
  - npm -v (This should show the installed version of npm)

- Create a new Next.js project using `create-next-app`.

3. Installing Next.js and Creating a New Project

Next.js requires Node.js and npm. To create a new Next.js project:

- Open your terminal.
- Run the following command to create a new Next.js project:
  npx create-next-app@latest **react-template**
- This command will create a new directory with the given project name, set up a Next.js project inside it, and install all necessary dependencies.

3.5 In the unlikely event of node/npm incompatibility

For Windows:

- Node.js: The best way to update Node.js on Windows is to download and install the newest version from the Node.js website.
- npm: Once Node.js is updated, you can update npm using the command line:
  - npm install -g npm

For macOS:

- Node.js: macOS users can update Node.js using a package manager like Homebrew. If Homebrew is not installed, it can be installed from https://brew.sh/. Once Homebrew is installed, update Node.js using:
  - brew update
  - brew upgrade node
- npm: Similar to Windows, npm can be updated with:
  - npm install -g npm

Using Node Version Managers

An alternative and popular method to manage Node.js versions is to use a version manager. This is especially useful if you need to switch between different Node.js versions for different projects.

- nvm (Node Version Manager): Works for both macOS and Linux. Windows users can use nvm-windows.
  - Installation and usage instructions for nvm are available at its GitHub repository: https://github.com/nvm-sh/nvm
  - For nvm-windows: https://github.com/coreybutler/nvm-windows
  - With nvm, you can install multiple versions of Node.js and switch between them easily.

4. Quick Walkthrough of the Project Structure

After creating the new Next.js project, navigate into your project directory:

cd **react-template**

Create Components directory

Create pages directory

Check out all the directories and pages in the project

2) React Basics in the Main Page (30 minutes)

- Understanding JSX/TSX (5 min):
  Explain syntax and how it differs from regular HTML.
  - Syntax is ostensibly the same but it is not technically HTML.
  - JSX and TSX are syntax extensions that allow developers to write HTML-like syntax in their JavaScript or TypeScript code. JSX is the original syntax extension used in React, while TSX combines JSX syntax with TypeScript's type system
- Creating Functional Components (10 min):
  - Import react
  - Create your first functional component that says “Hello World”
    - Show ability to move “export default”
    - Show Typescript types “const Home: React.FC = () => {}
    - Show ability to declare your own Types.
    - Explain how typing helps catch errors but is not too relevant to our current example
- Move around the primary div with scss.
  - Npm i sass
  - How is scss different than css?
    - Allows the ability to import variables. Show example from my website
    - Primary different is the specificity by nesting css selectors
  - Flexbox example
    - https://flexboxfroggy.com/ to learn
- Discuss props and how to pass data to components.
  - Props are how you pass data to components. They are simply parameters to a function a function that returns JSX (HTML)
  - The exact nuances of all prop passing details are little beyond the scope of this lecture.
  - Show a couple examples from other webpages
  - We won't really use them too much in the context of this lecture. But just know props are the parameters passed to functional components
  -
- State Management with Hooks (10 min):
  - Counter Toggler Examples
  - Other examples include setting user on a website
  - Setting which graph to show with a button

    - Many state examples are updated with user interaction onClick or user event.
    - Some scroll effect leads to some change in state

- The useEffect Hook (5 min):
    - Load the data with useEffect
    - useEffect can be linked with event like a window resize
    - Initiate effect when some value changes
    - Initiate timeout
    - Dom changes
    - Smooth scrolling effect

- Victory Charts in React App
    - npm install victory
    - To learn more about graphs: https://formidable.com/open-source/victory/docs/victory-chart/
    - Just copy
        - bar graph
        - Scatter graph
        - Line graph

3) Building a Multi-page Application with Next.js (30 minutes)

- Routing and Linking Pages (10 min)

- Show how to create new pages and navigate using the Link component.
    - Create navbar.tsx ⇒ Start with static text ⇒ move to link tags
    - Make sure app.globals.css are
    - Add navbar to Layout.tsx
    - Copy and paste _app.js
    - Copy and paste code into Layout.tsx
    - Create Footer.tsx
        - Add footer to Layout
- Dynamic Routes (10 min):
    - Modify the NavBar with Link tag for navigation
    - Reusing components vs building component with elements
        - Counter reuse in Page 2 vs Page 3 json render

---

4) Styling with SCSS (20 minutes)

- Basic SCSS walkthrough
    - Own 10-15 minutes styling your page the way you want, moving items around etc.
- Basic Styling with SCSS (10 min)

- Create and apply styles to components.
- Discuss nesting, variables, and mixins in SCSS.
    - Responsive Design Basics (5 min)
- Briefly touch upon media queries and mobile-first design principles.

5) Final Touches and Deployment (20 minutes)

- Review and Q&A (10 min)

- Recap the key points.
- Address any questions or confusions.

- Deploying the App (10 min):

- Introduce Vercel
- Demonstrate deploying the Next.js app.

- Wrapping it Together for your project
    - Use Flask API to render data from database
    - Use UseEffect to call data from json
    - Render into specific graph components
    - Overlay into your project