Build the Book Binge App - L000

Prerequisites: Set up your IDE and get familiar with React Foundations

Key technologies:

- **What is Node.js?**

Node.js is a server-side JavaScript runtime built on Chrome's V8 JavaScript engine. It provides a non-blocking, event-driven architecture.

- **What is *npm*?**

Node Package Manager is the default package manager for JavaScript that allows developers to install, share, and manage project dependencies.

- **What is React?**

React is a front-end framework for creating user interfaces (UIs). It allows developers to build complex UIs from small, isolated pieces of code called components.

- **What is JSX?**

JSX is a JavaScript syntax extension typically used with React to describe UI elements. It combines HTML with JavaScript logic to make code more readable and easier to write. It gets transpiled into plain JavaScript before running in a browser.

- **What is Webpack?**

Webpack is a module bundler for JavaScript applications that takes modules with dependencies and generates static assets that can run in a browser. This includes transpiling JSX for the browser. It is automatically installed and configured in React and Next applications.

- **What is Next.js?**

Next.js is a React framework that enables server-side rendering (SSR) and static site generation (SSG) for optimized and scalable web application development. SSR generates HTML on each request, while SSG generates HTML at build time.

If this is your first time doing front-end development on your local machine, you'll want to make sure you can edit and run a React application in your preferred IDE. If you're not sure which IDE to use, *VSCode* is a solid choice that works out of the box.

Before you can run a React project, you must have Node installed on your machine. To manage Node installations and easily change versions of Node across your projects, I recommend using *nvm* – Node Version Manager.

The *nvm* documentation gives clear instructions (**Mac & Linux/WSL**): https://github.com/nvm-sh/nvm?tab=readme-ov-file#intro

WINDOWS USERS: Install *nvm-windows* instead, following these instructions: https://learn.microsoft.com/en-us/windows/dev-environment/javascript/nodejs-on-windows

Clone the Book Binge project to your local machine: https://github.com/aehaldy/book-binge

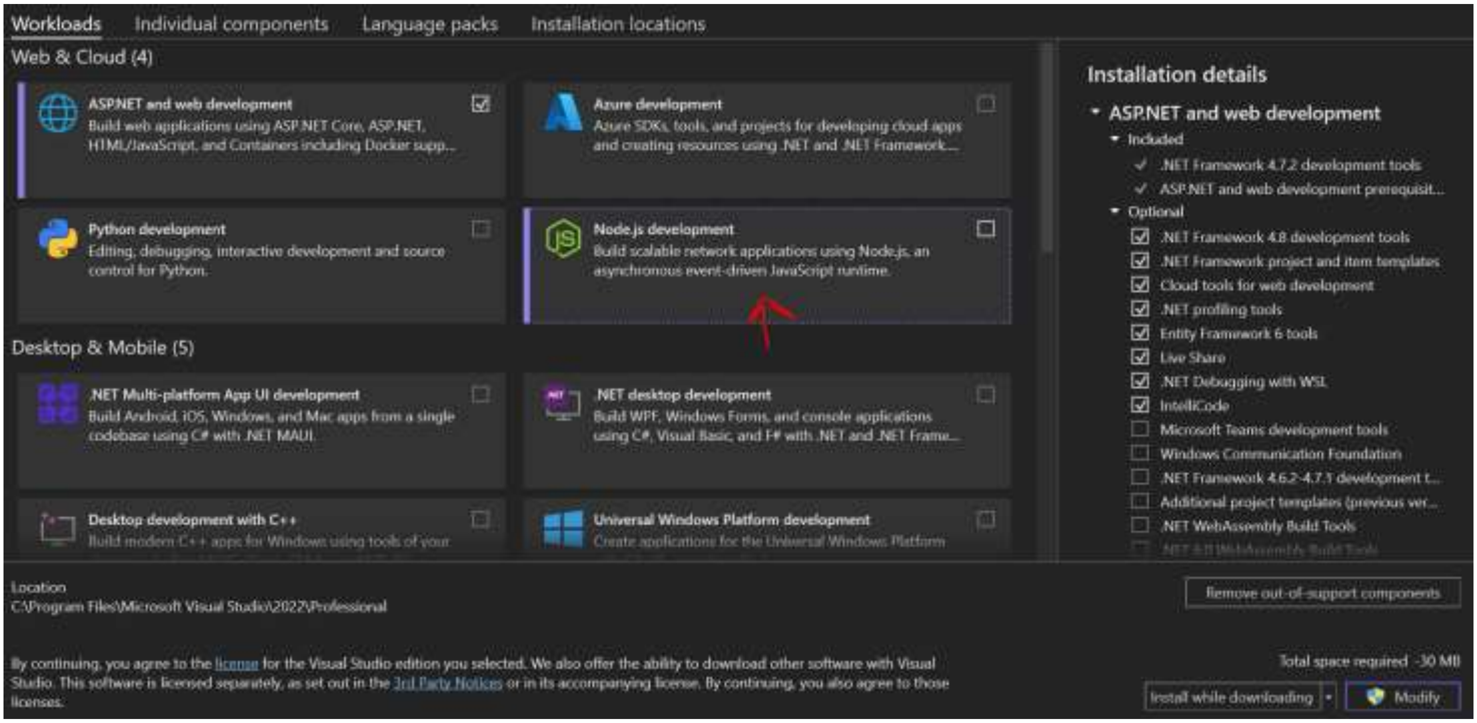
Use *nvm* to install Node **v18.17.0** to use in this project. Verify the version inside the project directory:

```
node --version
```

VSCode, as mentioned, works out of the box, but if you want to add some quality-of-life plugins, this guide gives a few suggestions, and introduces the debugger: https://www.jeffedmondson.dev/vscode-react-setup-2023/

Rider also works well with minimal effort. The official documentation from JetBrains shows how some of Rider's built-in features work in a React app: https://www.jetbrains.com/help/rider/react.html#starting_with_existing_react_application

Visual Studio is another option. You must install the Node.js development workload.
Go to **Tools → Get Tools and Features**, then choose the **Node.js development** workload and select **Modify**.



Web & Cloud (4)

☑ **ASP.NET and web development**
Build web applications using ASP.NET Core, ASP.NET, HTML/JavaScript, and Containers including Docker supp...

□ **Azure development**
Azure SDKs, tools, and projects for developing cloud apps and creating resources using .NET and .NET Framework...

□ **Python development**
Editing, debugging, interactive development and source control for Python.

□ **Node.js development**
Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

Desktop & Mobile (5)

□ **.NET Multi-platform App UI development**
Build Android, iOS, Windows, and Mac apps from a single codebase using C# with .NET MAUI.

□ **.NET desktop development**
Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET and .NET Frame...

□ **Desktop development with C++**
Build modern C++ apps for Windows using tools of your

□ **Universal Windows Platform development**
Create applications for the Universal Windows Platform

Installation details

- ASP.NET and web development
  - Included
    - ✓ .NET Framework 4.7.2 development tools
    - ✓ ASP.NET and web development prerequisit...
  - Optional
    - ☑ .NET Framework 4.8 development tools
    - ☑ .NET Framework project and item templates
    - ☑ Cloud tools for web development
    - ☑ .NET profiling tools
    - ☑ Entity Framework 6 tools
    - ☑ Live Share
    - ☑ .NET Debugging with WSL
    - ☑ IntelliCode
    - □ Microsoft Teams development tools
    - □ Windows Communication Foundation
    - □ .NET Framework 4.6.2-4.7.1 development t...
    - □ Additional project templates (previous ver...
    - □ .NET WebAssembly Build Tools

Location
C:\Program Files\Microsoft Visual Studio\2022\Professional

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Total space required -30 MB

Open the project as a folder (instead of a Solution).

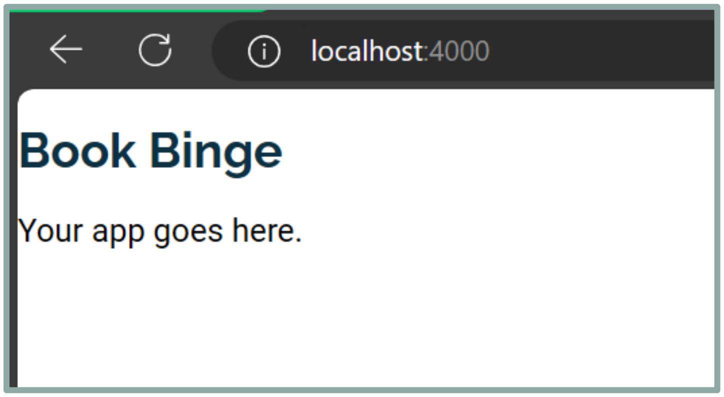
Once you have the project in your IDE of choice and have installed Node (via *nvm*), install the project's dependencies: (EDIT: contrast with **npm ci --legacy-peer-deps**)

```
npm install
```

Verify that the project runs locally:

```
npm run dev
```

Navigate your browser to *localhost:4000* to see the base application:



Book Binge

Your app goes here.

GitHub Copilot

GitHub copilot can be especially helpful when you are learning a new technology or working in an unfamiliar codebase. You may need to request access to Copilot from your ADM before you can finish installing the Copilot plugin in your IDE.

React Foundations

You should have a basic grasp of JavaScript/TypeScript, and of the foundational concepts presented in React's official first two *Learn React* chapters:

Describing the UI - https://react.dev/learn/describing-the-ui

Adding Interactivity - https://react.dev/learn/adding-interactivity

Review the concepts and feel free to skip areas you're already familiar with. Start with *Your First Component* and work through *Updating Arrays in State,* sixteen (short) lessons in total.

Additional resources:

JavaScript syntax cheatsheet - https://www.codecademy.com/learn/introduction-to-javascript/modules/learn-javascript-introduction/cheatsheet (5 min read)

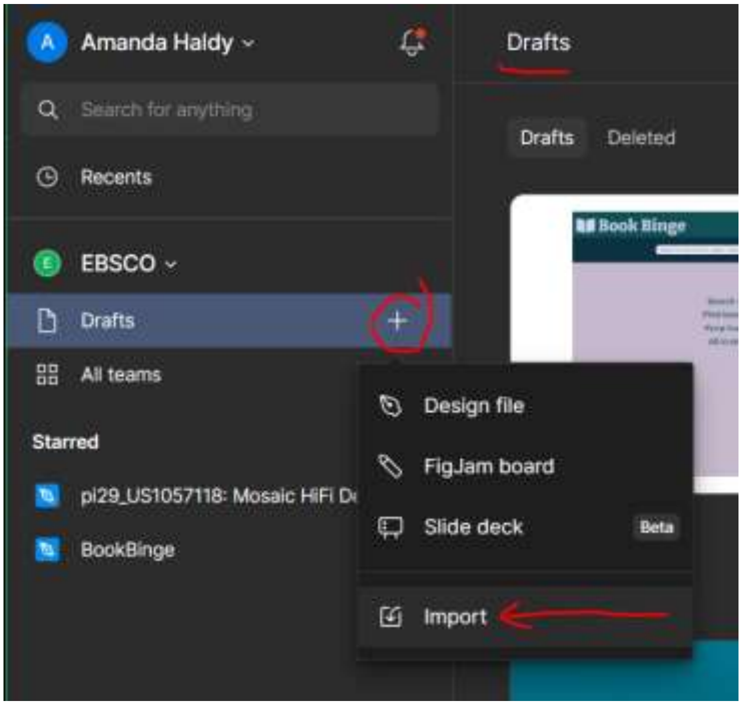
Typescript from a C# perspective - https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes-oop.html (5-10 min read)

Mozilla Docs - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects

Building the Book Binge app

To simulate a day-in-the-life of a front-end developer, this project comes complete with user stories reflecting Acceptance Criteria from the fictitious Book Binge Product team and a Figma design from a likewise imaginary UX Team. You will need to use the Figma client to inspect the UI's design and ascertain style choices such as fonts and colors. You can view the design at **TBD**

If the link is unavailable, just upload the Figma file (found in the repo at */Figma/BookBinge.fig*) to your personal Figma account for full access. A free Figma account is sufficient.



The project is already loaded with all necessary fonts and color variables so that you can focus on coding. In the */src* directory, you will find the full list of color variable names in the *global.scss* file.

UI Components

To simulate using 3rd-party or in-house UI libraries, this project comes with a small UI library you must leverage during development. You can inspect the available components by navigating to *localhost:4000/ui-library*.

Step-by-step lessons

Each lesson in the */BookBingeLessons* folder contains a pdf of instructions to walk through building the app at your own pace, and each lesson builds upon the previous one. Completed lesson content has a dedicated branch. As you start later lessons, feel free to branch off from the previous lesson's completed code to start fresh.

Happy coding!