SwiftHire- Resources, Documentation, Getting Started Guide
Sagar Sahu, Dhanush Ananthkar, Cole Augusta

Annotated Bibliography of Resources Used:

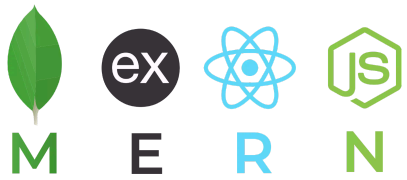
- YouTube Video Series- https://www.youtube.com/playlist?list=PL4cUxeGkcC9iJ_KkrkBZWZRHVwnzLIoUE
    - This tutorial introduces development using MERN with the following features:
        - Setting up an Express app on the backend
        - Designating API routes for simple requests to local server hosting
        - Connecting a MongoDB Atlas database to store json documents
        - Creating a model, schema, and controller for the selected data cluster of queries
        - Setting up a React app for frontend development and user interaction
        - Adding components for various web page options and user choices
- Comprehensive Online Guide- https://blog.nextideatech.com/how-to-get-started-with-the-mern-stack-a-comprehensive-guide/
    - This website provides in-depth understanding of installing and setting up the various technologies in the MERN stack, including:
        - Downloading the Node, MongoDB, and React packages
        - Testing server API requests, and experimenting with React web page components

Getting Started Guide:

- To start running backend server
    - Navigate to the root project directory: ‘cd …/backend’
    - Run ‘nodemon server.js’
- To start running frontend view
    - Navigate to the root project directory: ‘cd …/frontend’
    - Run ‘npm start’
- Basic features
    - Welcome page
    - Search job
    - Post a job
    - Sign in/up
    - About us page

Application Tech Stack/Documentation:

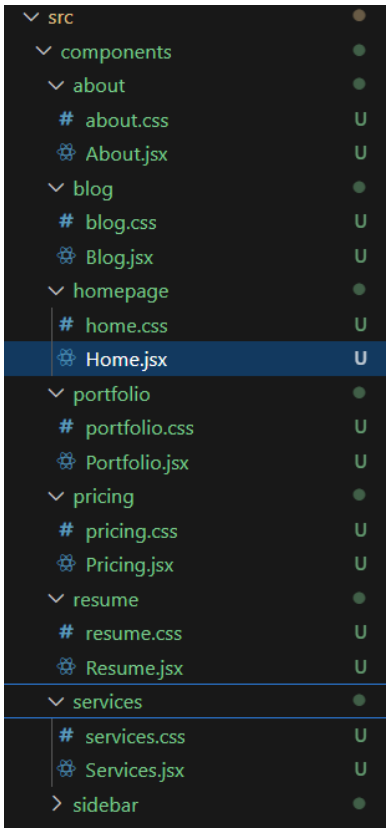
- Backend
    - Express.js
    - Node.js, nodemon, npm
    - MongoDB, mongoose, Atlas
    - Dotenv
    - Postman
- Frontend
    - React.js
    - HTML
    - CSS

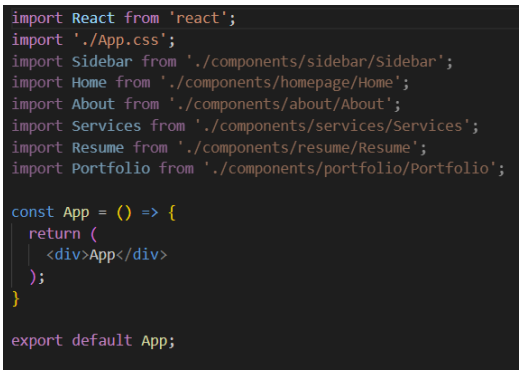


Learning React

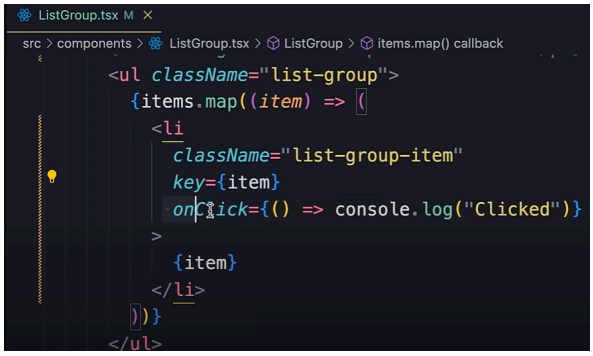
Here's what I learned from the React tutorial:

- **What is React?**
  - React is a JavaScript library used for building user interfaces. It lets you create reusable components that make web development easier and more efficient.
- **Setting Up the Dev Environment**
  - I installed Node.js, npm, and used `create-react-app` to quickly set up a React project. This tool auto-generates a basic project structure to get started fast.
- **Understanding Project Structure**
  - I got familiar with the key files (`index.js`, `App.js`) and how everything fits together in the `public` and `src` folders.
- **Creating React Components**
  - I learned to build functional components, which are the main building blocks of React apps. Components are reusable and independent, making development more modular.
- **How React Works**
  - React uses a virtual DOM to efficiently update the real DOM. It only changes what's necessary, which makes the app faster.
- **React Ecosystem**
  - React has an ecosystem of tools like React Router for navigation and libraries like Redux for managing global state.
- **Building Components**
  - I learned to build reusable components and how to pass data and behavior through `props`.





- **Fragments, Rendering Lists & Conditional Rendering**
    - Fragments: Helped me group elements without adding extra nodes to the DOM.
    - Rendering Lists: Learned to map over arrays to display lists of items dynamically.
    - Conditional Rendering: Show or hide elements based on conditions in the app (like using ternary operators).
- **Handling Events**
  - React makes it simple to handle events like clicks or form submissions with event listeners in JSX.



- **Managing State**
  - I used the `useState` hook to store and update data in a component. State changes trigger UI updates, keeping everything in sync.

- **Props vs. State**
    - Props: Passed down from parent components; they're read-only.
    - State: Managed within a component and can be updated dynamically.
- **Passing Children & Functions via Props**
  - I learned how to pass data, functions, and even child components through props to make components more flexible.
- **React Dev Tools**
  - The React Developer Tools extension is super helpful for inspecting components, their state, and props.
- **Exercises: Building Components:** I practiced building components like buttons and alert messages, which helped reinforce all the concepts.

Key Takeaways:

- React's component-based approach makes it easy to build dynamic, reusable UI elements.
- Learned to manage state, handle events, and pass data between components using props.
- Hands-on exercises gave me a better feel for React development.

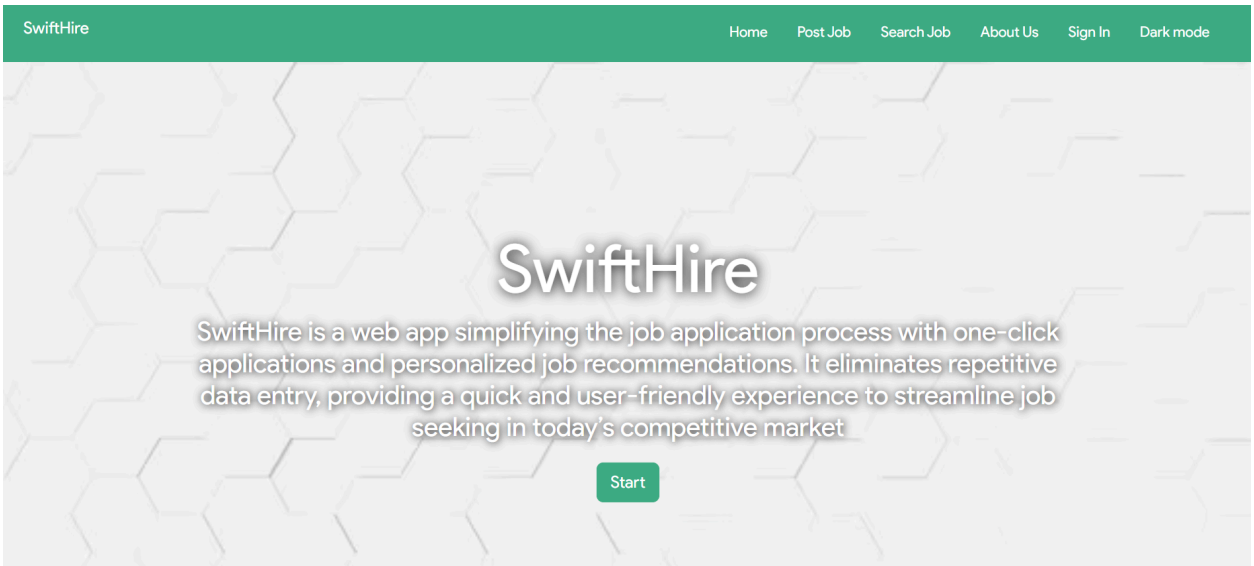


Figure 1: SwiftHire Main Welcome Page

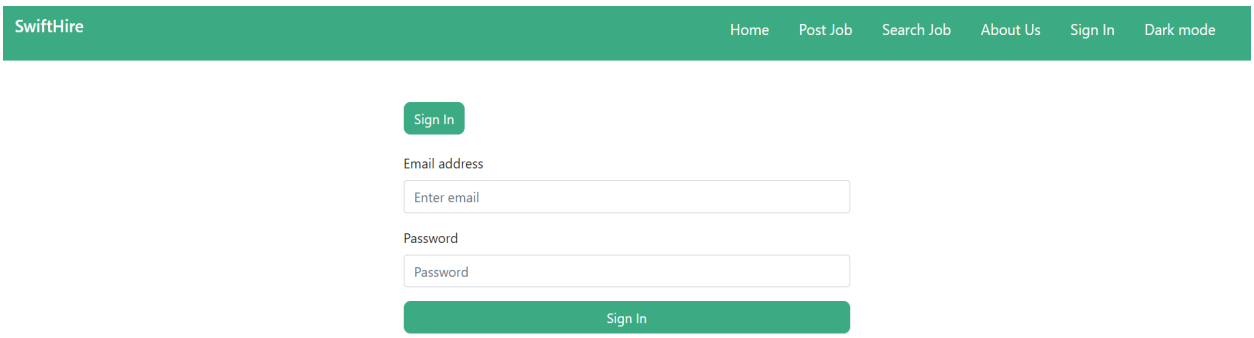


Figure 2: Sign In/Up User Page