**Day 1: Project Setup and Basic Structure**

1. Install Node.js and npm (if you haven't already) to manage your project dependencies.

2. Set up a new React project using Create React App (CRA) by running the following command in your terminal:

```

npx create-react-app my-portfolio

```

3. Once the project is created, navigate to the project directory:

```

cd my-portfolio

```

4. Explore the generated project structure and understand the basic files and folders.

**Day 2: Design and Component Creation**

1. Plan the design and layout of your portfolio. Sketch it out or use design tools like Figma or Adobe XD to create a visual representation.

2. Break down your design into smaller components.

3. Start creating React components based on your design.

4. Use CSS or a CSS-in-JS solution like styled-components or Emotion to style your components.

**Day 3: Content and Routing**

1. Define the sections and pages of your portfolio (e.g., Home, About, Projects, Contact).

2. Create placeholder content for each section.

3. Set up React Router to handle navigation between different sections and pages.

**Day 4: Data and API Integration**

1. If your portfolio requires dynamic data, set up a mock API using tools like JSON Server or use a static data file.

2. Fetch data from the API or import static data into your components.

3. Render the data dynamically in your components.

**Day 5: Enhancements and Responsive Design**

1. Add interactivity to your portfolio by incorporating animations, transitions, or other UI effects using libraries like React Spring or Framer Motion.

2. Test your portfolio on different devices and screen sizes to ensure responsiveness.

3. Use CSS media queries and responsive design techniques to make your portfolio adapt to different screen sizes.

**Day 6: Deployment and Optimization**

1. Build your React project for production by running the following command:

```

npm run build

```

2. Deploy your portfolio to a hosting platform of your choice (e.g., Netlify, Vercel, GitHub Pages).

3. Optimize your portfolio for performance by minifying and compressing assets, lazy-loading components, and implementing code-splitting techniques.

**Day 7: Final Touches and Testing**

1. Review your portfolio and make any necessary adjustments to improve the user experience.

2. Test your portfolio across different browsers to ensure cross-browser compatibility.

3. Validate your code using tools like ESLint or Prettier to maintain code quality and consistency.

4. Ask friends or colleagues to provide feedback and make any final revisions.

Remember, this is a general roadmap, and the timeline may vary depending on your familiarity with React and your design requirements. Feel free to adjust the roadmap to fit your needs and pace. Good luck with your portfolio project!